Sentiment Analysis of News Articles for Stock Price Prediction

By

Bolanle Esther Onifade
(beo01u)

School of Computer Science

University of Nottingham

Supervisor: Michel Valstar

Signature: ____________________________

Date: ____________________________

Abstract

Over the past few decades, many theories that describe the behaviour of the stock market have been proposed. These theories put forward ideas ranging from those claiming that the stock market cannot be predicted and to those claiming those that with the right assumptions, the stock market can be predicted. Regardless of this, many stock market prediction models (based on technical indicators or non-technical indicators) have been put forward. In this dissertation we aim to consider those non-technical factors that affect stock prices. These non-technical factors come in form of news articles. In the digital age of rapid response to ongoing situations, news articles about events tend to be released as soon as they occur and the assumption is that if we can track news sources, monitor them for the release of relevant news articles, we can use the sentiment orientation of the article (whether positive, negative or neutral) to predict the price of the stock market before the market has a chance to react to the article. This of course poses an important question on how the sentiment orientation of articles. There are several approaches that can be taken towards determining the class of articles but for the purposes of this dissertation, we will focus on the use of Support Vector Machines to perform classification of news articles. The output of classification will then be fed into a hybrid Support Vector Machines – Hidden Markov Model classifier to perform the price prediction.

Acknowledgements

I would like to firstly thank my supervisor Michel Valstar for the encouragement to explore areas of analysis that I otherwise wouldn't have thanked. Helping me think outside the box has helped take the dissertation much farther than I thought it would go. It would so be very poor form to neglect to thank Dr Robert Young who has helped me with aspects of finance and economics that I have found difficult to wrap my head around. I also would like to thank the few anonymous students from business school that helped with labelling the news articles.

To my parents, sisters and friends, I would like to say a massive thanks for all these years of supporting me through my education. A lot of time has passed since I started school 18 years ago and this dissertation is a representation of all the knowledge I have gained in that time. I truly appreciate all the support you've all expressed for me and this project.

Table of Contents

Table of Figures

Table of Algorithms

1. Introduction

One can arguably say that state of the world economy has been built with the stock market serving as a base. Hence, while there are other means of evaluating a country's economy, the state of the stock market is perhaps one of the more important means of doing so. Predicting the general direction of stock prices therefore is very important in order to make decisions regarding where and what kind of investment takes place. The current work attempts to predict the direction of movement of the (close) stock price of the next working day given the current day's information. We propose a prediction system which incorporates news articles other technical indicators such as the stochastic$\%K$, stochastic $\%D$.

The advent of the Internet brought with it improvements in many fields ranging from technologies that influence our daily lives to those that may one day take humans to other planets. One of the more mundane improvements is access to news articles – completely changing the way we now make decisions. We no longer have to wait until the following day to find out about events that took place today. Stock market traders are one group of people who rely heavily on this improvement on access to news. Often times, trades are executed using information from the news, unconsciously. This is because news articles need not necessarily carry extremely significant news in order to be usable to traders – news articles will often bear information about annual earnings, acquisitions and mergers, changes in administration and management as well as stock splits – and thus, news need not bear extremely catastrophic nor positive information, in order to be useful.

Traders will often base their trades on information from news articles regardless of how seemingly important it is. We therefore argue that the key to predicting the stock market is by monitoring extensive sources of news articles and extracting valuable information from the articles which can then we used to predict the stock market. The process of extracting information automatically from textual data is referred to as sentiment analysis. The current work therefore is an interdisciplinary work that ties together sentiment analysis, machine learning and finance for the prediction of the stock market.

The next section details the motivation for this project. We then carry on describing our objections, the assumptions we make and structure of the current work.

1.1. Motivation

A lot of past work has been done using technical and fundamental indicators such as the current investment, general economy, recessional periods, currency and industry – this is known as fundamental analysis. However, very few articles have attempted to go beyond that and while some of these methods perform reasonably well, very few articles in the literature have attempted to go beyond historical data. Even still, within the set of articles that attempt to use recent (external) information such as news articles, very few articles attempt to incorporate historical technical data. With this work, we aim to further fill in this gap by utilising both historical data (section 3.6.)that provide us with an indication of how well the stock price has done in the past as well as current news that provide us with a sense of what the general sentiment regarding a certain stock is. This is primarily because it stands to reason that we will make much higher profits with finding the middle ground between the two extremes. A few related works have attempted to use both methods to predict the stock market (Deng, Mitsubuchi, Shioda, Shimada, & Sakurai, 2011); however, the backbone of the current proposed model uses a Support Vector Machine and Hidden Markov Model (SVM-HMM) hybrid model for the predictions – which to our knowledge hasn't been used previously.

1.2. Objectives

The primary goals are to propose a system by which the stock price can be predicted as well as perform experiments that test the performance of the system. We also explore the effectiveness of the hybrid system in predicting stock prices.

Furthermore, we aim to extensively and conclusively review related, previous work on the prediction of the stock market both technical indicators and sentiment-based indicators. Although our discussion of sentiment analysis is heavily favoured towards machine learning based techniques, we aim to give a well-rounded discussion by taking non-machine learning based techniques into consideration.

1.3. Assumptions

There are several schools of thought regarding whether the stock market can at all be predicted and in order to continue, we must first discuss the current hypotheses and highlight which hypotheses we have based the project on. Our hypotheses come in the form of the three levels of the Efficient Market Hypothesis.

The weak-form efficient market hypothesis assumes that the market is efficient. In addition, it also assumes that the rates of return are independent, meaning that past return has no bearing on future return. Following from this, traders, both algorithmic and human, make invalid assumptions when trading.

The semi-strong form efficient market hypothesis assumes that the market, at all times reflects all publicly available information. The stock market hence responds very quickly to new information. Conclusively, this implies that potential investors cannot profit on the stock market as investors can only trade based on new information, after the market has adjusted to it.

The strong form efficient market hypothesis assumes that the market, at all moments reflects both publicly and privately available information. This incorporates both the weak and semi-strong form of the efficient market hypothesis and following this hypothesis, no one can make money.

Hence, it's quite clear that one of the first decisions to be made is whether or not the efficient market hypothesis is one of our assumptions and if it is, which level. It's quite clear that to assume at all the any form of the Efficient Market Hypothesis would invalidate the current work; hence, we do not assume the efficient market hypothesis. The Efficient Market Hypothesis assumes that all investors are rational and that there exists a perfect flow of information which clearly is invalid. Instead, we assume that at the end of each working day, the stock price reflects the available information. It's safe to make this assumption as should an important piece of news be released that alters the stock price significantly, the price at the end of the day will reflect this. Should a company be, say, downgraded by standard and poor, the stock price will reflect this downgrade until the rating is increased once more. That is to say that the *shockwave* of a downgrade isn't just absorbed by the market – the stock price trend line of such a company will continuously, visibly be a *symptom* of the downgrade.

We assume a much simpler model of the stock market – predicting the direction of movement of the close price of the stock market rather than the intra-day price. It's important to note at this point that predicting the stock market needn't necessarily involve predicting the exact values – predicting simply the direction of movement is enough (Elkan, 1999; Lavrenko, Schmill, Lawrie, Ogilvie, Jensen, & Allan, 2000; Thomas & Sycara, 2000).

We also assume that the effect of information extends into multiple working days. Hence, an Hidden Markov Model can identify patterns based on the trend of the past several working days.

Finally, we make the assumption that any important trend that changes the rate of return of a stock price in any significant way can be extracted from news articles.

1.4. Organisation of Document

In the succeeding chapter, we provide an in-depth review of the literature both on sentiment analysis and on stock price prediction. In Chapter 3, as we assume that the average reader is unfamiliar with the techniques used for the project, we provide preliminary background knowledge. Chapter 4 details the scientific method used for the project. Chapter 5 details the results of the experiments run. To conclude, chapter 6 details the closing remarks.

2. A Brief Review of the Literature

The current chapter is split into three sections. The first section reviews the recent work on sentiment analysis. The second section reviews the work on specifically, the domain of stock market prediction with sentiment analysis. We note that there are several means by which sentiment analysis can be applied for use in stock price prediction and we aim to at least touch on the more interesting methods in the literature. Finally, we provide concluding remarks on the review.

2.1. Sentiment Analysis

Bing Liu (2012) highlights three levels of sentiment analysis: document based, sentence based and aspect-entity based. Document-based sentiment analysis pertains to the classification of an entire document and the key assumption is that the entire document focuses on a single entity. Sentence-based analysis is more in-depth analysis of individual sentences. Aspect-entity based analysis focuses on determining the subject of discussion as well as the sentiment. Even further, opinions are split into regular and comparative opinions – regular opinion express opinion on a single entity while comparative opinion expresses opinion on two or more entities.

Bing Liu highlights that the most important indicator of sentiment are sentiment words such as *poor, happy, sad, brilliant* or *excellent.* Of course, one glaring issue with simply analysing based on sentimental words is the use of sarcasm in language. This, in addition with the fact that sentimental words can change orientation depending on the context means that sentiment analysis can't be reduced simply to a keyword search. It should be noted at this point that financial news articles generally do not benefit from the use of sentimental words. Liu discusses extensively sentiment analysis using varying techniques but since we are simply interested in only document-based classification, we focus on articles related to such classification. It is recommended that any reader who wishes to gain a more complete and up-to-date overview of the current sentiment analysis methods should refer to Bing Liu's work on the subject.

One of the more important aspects of document-based classification is the determination of features. Popular ones include terms and their frequency (which is the selected features for this current projected), parts of speech (POS) – introduced by Turney (2002) – and their tags such as sentiment words and sentiment flippers (words that change the orientation of sentiment words such as *not*). Bing Liu also comments that some domains are easier than others, for example movie reviews tend to be easier to analyse than car reviews – this is due to the multi-entity nature of car reviews. Compare:

S1*:* The movie is utterly captivating

S2: While I like the leather seats, the gear is a bit hard to manipulate

In the literature, there are two main approaches to feature generation: a lexicon-based approach and a bag-of-words approach. The lexicon-based approach as used in Whitelaw et al. (2005) typically involves a set of example words which are then used as a seed for building a larger lexicon through the identification of synonyms in a semi-automatic manner. Bag of words approach typically involves the representation of words as a numerical value indicating their presence, frequency or term frequency-inverse document frequency (TF-IDF)score Yong et al. (2011). Please refer to section 3.1.1 for our discussion of TF-IDF. One of the more often cited criticisms of the bag-of-words approach is the loss of semantic association between terms.

Work has been done to improve on the TF-IDF by introducing the delta TF-IDF (Martineau & Finin, 2009). Delta TF-IDF improves the importance placed on words that are unevenly distributed in the corpus and discounts those that are evenly distributed – the rationale being that the less unevenly a feature is represented, the higher the chances that it's relevant for its classification. Delta TF-IDF is formulated mathematically as:

$$V_{t,d} = C_{t,d} * \log_2\left(\frac{|P|}{P_t}\right) - C_{t,d} * \log_2\left(\frac{|N|}{N_t}\right)$$

$$= C_{t,d} * \log_2\left(\left(\frac{|P|}{P_t}\right) * \left(\frac{N_t}{|N|}\right)\right)$$

$$= C_{t,d} * \log_2\left(\frac{N_t}{P_t}\right) \quad \textbf{(2.1)}$$

where $C_{t,d}$ is the number of times term t appears in document d, $P_t$is the number of positively labelled documents containing term t, $|P|$ is the number of positively labelled documents, N_t is the number of negatively labelled documents with term t,$|N|$ is the number of negatively labelled documents and $V_{t,d}$ is the value for term t in document d. Using feature values derived by delta TF-IDF on movie review classification, a SVM achieved an accuracy of 88.1% compared to traditional TF-IDF based classification accuracy of 82.85%.

Feature selection is critical in sentiment analysis as corpuses tend to be polluted with noise, reducing the performance of any developed system. Core techniques involved in feature selection are stop-word removal and stemming. Furthermore, statistical methods such as the point-wise mutual information (Turney & Littman, 2003; Wilson, Wiebe, & Hoffmann, 2005) and chi-square (χ^2) are used in the feature selection process.

The point-wise mutual information $PMI_c(t)$ measures the correlation between a term t and the class c. Assuming mutual independence, PMI can be calculated using the joint distribution and the individual distributions. This is formulated mathematically by Aggarwal and ChengXiang (2012) as:

$$PMI_c(t) = \log\left(\frac{F(t) \cdot p_c(t)}{F(t) \cdot P_c}\right) = \log\left(\frac{p_c(t)}{P_c}\right) \quad \textbf{(2.2)}$$

where the expected co-occurrence of c and w is $F(t) \cdot P_c$ and the actual co-occurrence is $F(t) \cdot p_c(t)$. A PMI of greater than 0 means a positive correlation between w and c and the inverse is the case for a PMI less than 0. We delegate to section 3.1.3 our explanation of χ^2.

Sentiment classification techniques are split into two broad range of techniques: machine learning techniques (Yong, Xu, & Ren, 2011; Pang, Lee, & Vaithyanathan, 2002; Kang, Yoo, & Han, 2012; Kaufmann, 2012; Moraes, Valiati, & Neto, 2013), lexicon-based approach – which is further split into dictionary-based approach (Guang, Xiaofei, Feng, Yuan, Jiajun, & Chun, 2010; Minging & Bing, 2004; Kim & Hovy, 2004) and corpus-based approach (Deerwester, Dumais, Landauer, Furnas, & Harshman, 1990). We will focus on the machine learning techniques.

Pang et al. (Pang, Lee, & Vaithyanathan, 2002) using a corpus of reviews that were classified according the number of stars associated with the text reviews, applied naïve bayes, maximum entropy and support vector machines. They compared the performance of unigrams, unigrams + bigrams, bigrams, unigrams + POS, adjectives, top unigrams and unigrams + positions using three-fold cross-validation. The performance of each variation in feature depended heavily on the type of technique used. However SVMs on average performed better than both naïve bayes and maximum entropy. It's important to note that the classifiers performed best when the terms were represented simply as a presence as opposed to frequency. Unigrams performed best in their experiments using the Support Vector Machine-based model with an accuracy of 82.9%.

Yong et al. (2011) propose a method of sentiment analysis similar to the approach the current work has taken, the pre-process the words by removing stop words and tokenisation. Each word is then sorted based on calculated mutual information and select a number of words as the feature item and classify using an SVM. They neglect to specify the domain and source of the corpus. However, they achieve very high classification rates with total precision of 81.11%, recall of 81.42% and F-value of 81.25 in a closed test.

2.2. Stock Price Prediction Using Sentiment Analysis

Often, it is the case that a useable corpus of financial news isn't available hence authors are forced to generate their own data (Zhan, Cohen, & Atreya) either by manually classifying or automatically generating data (Fung, Yu, & Lu, 2005; Zhan, Cohen, & Atreya). Zhan et al. in their comparison of manual and automatically generated data highlight that the key to classifying news articles manually is the general information that is being conveyed by the articles. Mergers, lower interest rates are general considered *good news* while corruption, lawsuits, wars are considered *bad news.* Working with automatically generated data based on simply the stock price movement (if the stock price goes up after the article is released then the article is good otherwise, it's bad). The classifier developed has F1-scores of 0.26, 0.38 and 0.36 for positive, neutral and negative articles respectively. They attributed the poor performance to the lack of data and poor article selection whilst also suggesting that news articles are better suited to long-term prediction as opposed to day-to-day prediction. It's quite clear that there are while there are issues with automatic generation of data (section 5.3.2.), the method employed by Zhan et al. can be said to be too naïve.

Regardless of the criticisms of automatically labelling news article based on the direction of movement of the next day's price movement, Kaya et al had a 60% accuracy using the method as well as X^2-based feature selection. Fung et al. (2005) utilise piecewise linear segmentation to automatically classify news articles – a method that has been employed in the current work and will be explained in the next chapter.

Gidofalvi(2001), derive a unique method of automatically assigning labels to news articles by aligning news articles to the intraday stock data and although it doesn't perform very well, we spend some time on it due to its contribution to the literature. A window of influence is defined which is used to evaluate the possible effect of a news article. The author defines the window of influence of an article d with the timestamp t as the lower boundary offset and the upper boundary offset from t. An offset is negative is $t + offset$ is prior to t. In addition, news articles that aren't published within the opening and closing market times are filtered out as these are said to be ambiguous.

To establish how stable/volatile a stock is, a β-value is calculated using the linear regression on data-points (Δ index-price, Δ stock-price). Hence, a β-value of 1 means that whenever the index price changes by δ, the stock price is expected to change by δ as well. A β-value of 2 means that whenever the index price changes by δ, the stock price is expected to change by 2δ as well. A β-value of greater than 1 are relatively volatile and the inverse is the case for stocks less than 1.

In order to remove the effects of the exponential change in price, the formula is:

$$\Delta\mathrm{price}(u, v) = \ln\frac{\mathrm{price}(u)}{\mathrm{price}(v)} \tag{2.3}$$

The movement of a stock within a time interval is:

$$\mathrm{m}(u, v) = \frac{\Delta\mathrm{sp}(u, v\,)}{\beta} - \Delta\mathrm{ip}(u, v) \tag{2.4}$$

where $\Delta\mathrm{sp}(u, v\,)$ is the change in the stock price and $\Delta\mathrm{ip}(u, v)$ is the change in index price during the time interval $[u, v]$ A news article d with timestamp t can then be measured with offsets $[l, u]$ to receive a score of $\mathrm{m}(t + l, t + u)$.

Movement classes can then be defined from these equations:

$$mc(m) = \begin{cases} UP & if\ m > P_{positive} \\ DOWN & if\ m < P_{negative} \\ EXPECTED & otherwise \end{cases} \tag{2.5}$$

Where $P_{positive}$ and $P_{negative}$ are threshold values. Naïve Bayesian can then be used to predict the probability of a document belonging to a class.

The predictive power of the classification/system discussed is low with the system performing worse than randomness. On analysis, low r^2 values show that the movement measure model is poor-fitting to the stock price.

On evaluating the labelling of the news articles, they discover the highest statistically significant settings of $p_{negative} = -0.002$ and $p_{positive} = 0.0002$. The authors also find that the most statistically significant settings for alignments are [-20, 0] and [0, 20], that is 20 minutes before and 20 minutes after the release of the news article.

The predictive capability of the classifier was very low and the apparent reasoning for this the $\beta - values$ do not accurately model the relative movement of the stock correctly. Regardless of the accuracy, they concluded by acknowledging that their results contradict the efficient market hypothesis.

Perhaps, more unconventionally, is the use of tweets from twitter to predict the stock market (Bollen, Mao, & Zeng, 2011). They determine the correlation between the mood of the twitter feeds and the Dow Jones Industrial Average. The moods are determined using OpinionFinder (classifies into positive and negative) and Google-Profile of Mood States (GPOMS). A Granger Causality Analysis and Self-organising Fuzzy Neural Network are then used to determine the validity of the hypotheses that moods predict the stock market. The orientation provided by OpinionFinder is discovered to be less predictive than the GPOMS dimension *Calm*. They also claim that there exists a (3-4 days) time lag between the mood expressed on twitter and the changes in the DJIA values – hence stock price movements can be known well in advance.

Bar-Haim et al. (2011) propose a method of identifying expert investors from twitter feeds, which can then act as a basis for predicting the increase in stock prices. They compare two extreme methods: focusing on tweets that explicitly state transaction details as well as learning the correlation between the stock price and the tweet's contents. The second approach removes restrictions on the applicable tweets but with the caveat that a lot of noise is likely to be introduced into the training process. They show that making the process user-sensitive improves the prediction accuracy. The algorithm involves a classifier which classifies a time-annotated set of tweets by each user and classifies each as bullish, bearish or neutral. Each tweet can then be evaluated for correctness by determining that the stock market behaves in accordance with the classification of the tweet. Tweets finally are ranked according to correctness. Another method utilised is unsupervised learning based on the timestamp of the tweet. Using several methods: joint-all model (a single SVM model trained on all tweets), transaction model (finds expert users in based on the correlation of their tweets to the movement of the stock price), per-user model (removes noise by unsupervised learning for each potential expert), joint-experts model (using the per-user model, train a single SVM model). They conclude that the most accurate models are the per-user and the joint-experts perform the best.

Zhang and Skiena (2010) compare blogs and news as basis for prediction, perform large-scale analysis of the stock market and propose a trading strategy based on sentiment data. Data from Dailies (an aggregator of news), twitter, Spinn3r RSS feeds and LiveJournal was processed by Lydia (a text processing system), resulting in time series consisting of a time series of words and their orientation. They discovered that the media exposure correlates more to the stock market of certain industries (Aerospace and Defence) and less so for others (Software and Computer Services).

AZFinText (Schumaker & Chen, 2005) works with the assumption that 20 minutes after a news article is released, the stock price reflects the effect of the news. As opposed to labelling with a polarity (up or down), it labels using the stock price 20 minutes after the news article is published. The system uses proper nouns as the features and filters the selected features by only further selecting proper nouns that occur three or more times. It uses support vector regression to try to predict the price of the stock in 20 minutes. For a period of 23 trading day, using 2809 articles and predicting for the S&P 500 Index, they achieved a 8.50% return on investment while the S&P 500 Index achieved only a 5.62%.

NewsCATS (Mittermayer & Knolmayer, 2006) categorises news articles into "good", "bad" and "no movers". Using a thesaurus as a curator, they classify press releases as good if the maximum gain in 15 minutes after its release is large (> 3%) and the maximum loss is (< 3%). The inverse is the case for "bad" news. Neutral news are classified as those whose gain and loss do not exceed 3%. NewsCATS achieved an average of 0.29% over 2602 trades while a simulated random trader achieved a profit of 0.07% over 2599 trades. Mittermayer and Knolmayer conclude by cautioning that the results do not factor transaction cost and hence do not accurately reflect the potential profit.

2.3. Concluding Remarks

Sentiment analysis has been identified to have a broad range of applications from determining the public sentiment of films to the current application of stock price prediction. It's quite clear that the problem of sentiment analysis faces different challenges depending on the domain or the dataset source. Our review of the literature has been heavily biased towards techniques that are specifically applied to this project and readers are encouraged to read the survey by Medha et al. (2014) for more complete overview of the domain. In addition, another reasonable conclusion from the literature review is that sentiment analysis is not only a reasonable, quick method of text mining but also can be applied as a step in more complex processes such as stock price prediction.

This concludes our review of the literature on sentiment analysis for stock price prediction.

3. Background Knowledge

In the previous chapter, we introduced in passing several key concepts, especially in the context of sentiment analysis. In this chapter, we aim to arm the reader with the key knowledge required to gain an understanding of the rest of this document. In addition, some concepts that were mentioned in the previous chapter are fully explained here. Hence, this chapter (especially section 3.1.) is also a good precursor to chapter 2.

3.1. Sentiment Analysis

Sentiment analysis, according to Wikipedia[1], is defined as "*the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials*." We have seen in chapter 2 that the possibilities in terms of approaches to the general problem of sentiment analysis are extremely varied; hence, once more we focus on the steps required to a machine learning-based solution. We split our discussion into clear steps that one would expect when performing a machine learning task.

3.1.1. Feature Generation

Regardless of whether the chosen approach to acquiring labelled is manual or automatic, feature generation from the corpus[2] has to occur. Tokenising is the preferred approach to feature generation. Tokenisation refers to the splitting of a document into single words, phrases, nouns or other parts of speech which are called tokens. Tokens often disregard any form of punctuation. These tokens usually come in form of unigrams, bigrams, trigrams and other n-grams.

Unigrams are n-grams that are of size one. For illustrative purposes, we show the feature generation from a document consisting of only one sentence.

D1: The fat cat sat on the fat dog.

$\widehat{\boldsymbol{D1}}$**:** The, fat, cat, sat, on, the, dog

Higher-order n-grams were introduced to solve the issue of complete disintegration of any semantic relationship between the terms; unigrams perform quite well, despite this criticism (Pang, Lee, & Vaithyanathan, 2002).

Bigrams are n-grams that are of size two. Bigrams introduce a relationship between the individual words:

D2: Standard and Poor changes Goldman recommendation from A to BBB

$\widehat{\boldsymbol{D2}}$**:** Standard and, and Poor, Poor changes, changes Goldman, Goldman recommendation, recommendation from, from A, A to, to BBB.

In the case of the D2, we can see why bigrams can be beneficial: a classifier might be able to identify the negative orientation of the document due to the bigrams *from A* and *to BBB*. Conversely a classifier might be able to detect the positivity should the bigrams be *from BBB* and *to A*. As far as unigram-based classifiers are concerned, there isn't any difference. While one

[1] Definition acquired from: *en.wikipedia.org/wiki/Sentiment_analysis*
[2] Wikipedia (*en.wikipedia.org/wiki/Text_corpus*) defines a corpus as a large and structured set of texts.

might think that the higher the order of the n-gram, the better a classifier would be, this isn't the case. Higher-order n-grams bear more similarity to the source document than an overall pattern; in short, higher-order n-grams become less useful in determining sentiment orientation.

N-grams are not the only means of generating features. Features can also consist of noun phrases or pronouns. These are usually accompanied by other parts of speech like adjectives. In order to extract nouns, documents have to be processed by a part of speech tagger such as the one provided by Stanford[3]. Even more specific parts of speech such as Proper Nouns or Name Entities[4] can act as features.

As part of feature are the key steps of stop-word removal and stemming. Stop-words are words in the document that occur very frequently across the corpus and often bear no weight in the orientation. Examples of such words in the English language are *a*, *the*, *rather*, *but*, *also*. Textfixer provides a full list of such words[5]. Removal of these words is key to reduction of noise across the corpus. In some cases, stop words are key to the determining the orientation of sentences (often relevant at sentence-level classification). For illustrative purposes, the following sentence shows the case in which stop words are useful:

D3: If only Mr Tim was a decent actor.

The removal of the stop-words *if* and *only* completely alter the sentiment orientation of the sentence.

Stemming refers to the linguistic normalisation of words from their inflected form to a common form. Words needn't be mapped to its exact morphological stem, a relation is usually all that's needed. The example below shows the effect of stemming.

Cooperation, Cooperating, Cooperative, Cooperates $\mapsto$ Cooperate

Popular stemming algorithms are the Porter Stemmer and the Snowball Stemming Algorithms. Like stop-word removal, stemming reduces the noise in the features and allows for better performance.

3.1.2. Feature Representation

In order to make the documents useful to numeric classifiers such as the neural network or support vector machine, the terms need to be transformed into their weighted forms. The weighting of the terms is shown to be at least as important as their selection (Strzalkowski, 1994).

The vector space model introduced by Salton et al. (1975) is the preferred algebraic method of representing textual documents. The document is represented as a single vector where each dimension in the vector is a single feature. Features that do not exist in a particular document *d*

[3] The Stanford Natural Language Processing Group provide a copy of their tagger at nlp.standford.edu/softwaretagger.shtml
[4] Name entities are similar to proper nouns but refer to specific entities. Dates, organisations, places or numerical data can serve as name entities.
[5] Common stop words in the English Language at www.textfixer.com/resources/common-english-words.txt

are simply given a value of 0. The weights are usually calculated using simply their presence (binary), counts (integer), frequency(float) or their term frequency inverse document frequency (TF-IDF) score (float). Hence we can say that in a corpus of $|D|$ documents and m features, a document d_j is represented as:

$$d_j = \left(w_{1,j}, w_{2,j}, w_{3,j}, \dots, w_{m,j}\right) \tag{3.1}$$

where $1 \leq j \leq |D|$. We described an improvement on TF-IDF in chapter 2 and now, we provide the mathematics behind the traditional TF-IDF (Salton, Wong, & Yang, 1975). Given, the document$j$, the weight $w_{t,d}$ is calculated as:

$$w_{t,d} = tf_{t,d} \cdot \log\left(\frac{|D|}{1 + |\{d' \epsilon\, D \mid t \,\epsilon\, d'\}|}\right) \tag{3.2}$$

where $|\{d' \epsilon\, D \mid t \,\epsilon\, d'\}|$ is the number focuments which contain the term t. 1 is added to prevent divide-by-zero errors. $tf_{t,d}$ is the term frequency and is given by the formula:

$$tf_{t,d} = \frac{n_{t,d}}{\sum_m n_{m,d}} \tag{3.3}$$

where $n_{t,d}$ is the number of times feature f_t occurs in document d_j and $\sum_m n_{m,d}$ is the total number of terms in the document d_j. The higher the value assigned to a feature, the importance, it is given.

3.1.3. Feature Reduction

Due to the incredibly large number of features generated by tokenisation, it's necessary for feature reduction to take place; the preferred methods for this are χ^2 and singular value decomposition (Golug & Kahan, 1965), although we only discuss χ^2 based feature reduction.

χ^2 is used to test the level of independence between the features and the classes. A χ^2 value of 0 indicate a lack of dependence while a large value implies a large dependence. Features with a χ^2 value greater than a set threshold are selected. χ^2 is formulated mathematically by Aggarwal and ChengXiang (2012)as:

$$\chi_c^2(t) = \frac{n \cdot F(t) \cdot (p_c(t) - P_c)^2}{F(t) \cdot \left(1 - F(t)\right) \cdot P_c \cdot (1 - P_c)} \tag{3.4}$$

where n is the number of documents in the corpus, $p_c(t)$ is the conditional probability of documents being assigned to label c, given that they contain t, P_c is the number of documents assigned label c. and $F(t)$ is the number of documents which contain word t.

3.2. T-Test Based Split-and-Merge Piecewise Linear Approximation

In their paper, The Predicting Power of Textual Information on Financial Markets, Fung et al, (2005) present a means of generating labelled data by detecting trends in a time series and aligning news with the trends; we present their algorithm here.

3.2.1. The Splitting Phase

The split-phase of the algorithm handles discovering the trends in a time series while the merge phase helps avoid over-segmentation. Each time series can be represented as a list of tuples – each tuple containing the price and time:

$$T = \{(t_0, p_0), (t_1, p_1), \dots, (t_n, p_n)\} \tag{3.5}$$

where p_i is the price at t_i for $i \in [0, n]$ and the time series T has a length n.

The process starts by representing the trend with a line L defined by the first and last points (a segment). In order to decide if the line is enough to represent T, a one-tail t-test is defined:

$$\begin{gathered} H_0: \varepsilon = 0 \\ H_1: \varepsilon > 0 \end{gathered} \tag{3.6}$$

where ε is defined as the expected mean square error:

$$\varepsilon = \frac{1}{k} \cdot \sum_{i=0}^{k} (p_i - \widehat{p_i})^2 \tag{3.7}$$

where k is the number of points in the segment. $\widehat{p_i}$ is the projected price (derived from the line L) at time t_i. The t-statistic is defined by:

$$t = \frac{\varepsilon}{\sqrt{\frac{\hat{\sigma}^2}{n}}} \tag{3.8}$$

Where $\hat{\sigma}^2$ is the standard deviation. The t-statistic is compared with the t-distribution using a $n - 1$ degree of freedom and an $\alpha = 0.05$. Hence, there is a 0.05 probability of the null hypothesis H_0 being accepted given that is incorrect. If H_0 is accepted, then the mean-squared error is low and the projected trend line L is very similar to the actual time series T. If $H_1$is accepted then, the line L is not enough to represent T. If H_1 is accepted, then the line is split where the error norm is maximum – $\max_i\{(p_i - \widehat{p_i})^2\}$ – resulting in two segments. The process iterated by repeating the process on the two segments. The algorithm is represented in algorithm 3.1, as produced by Fung et al.

3.2.2. The Merging Phase

Over-segmentation is the existence of adjacent two segments whose slopes bear enough similarity to not warrant two separate segments. To fix this issue, the merging phase merges these segments into one large one. The merging phase aims at combining adjacent segment whose merging would not result in H_0 being rejected.

Consider a time series $T_{temp} = \{(t'_0, p'_0), (t'_1, p'_1), \dots, (t'_m, p'_m)\}$ (where $m \ll n$) , the result of splitting time series T. $S_i = \{(t'_i, p'_i), (t'_{i+1}, p'_{i+1})\}$ is a segment in T_{temp}. Should potential merge of the segments S_i and S_{i+1} not result in H_0 being rejected then they are regarded as candidates for merging. Let the list L_{merge} contain all such pairs of candidates. The selected candidates are the pair that would result in the lest increase in ε. This process is repeated until the t-test is rejected. Hence $L_{merge} = \emptyset$. The merge algorithm is provided in figure 3.2.

split($T[t_a, t_b]$) – split a time series of T of length n from time t_a to time t_b where $0 \leq a < b \leq n$

$T_{temp} = \emptyset$
$\varepsilon_{min} = \infty$
$\varepsilon_{total} = 0$
for $i = a$ to b **do**
  $\varepsilon_{min} = (p_i - \widehat{p_i})^2$
  if $\varepsilon_{min} > \varepsilon_i$ **then**
    $\varepsilon_{min} = \varepsilon_i$
    $t_k = t_i$
  end if
  $\varepsilon_{total} = \varepsilon_{total} + \varepsilon_i$
end for
$\varepsilon = \varepsilon_{total} + \varepsilon_i$
if t-test.reject(ε) **then**
  $T_{temp} = T_{temp} \cup$ split($T[t_a, t_k]$)
  $T_{temp} = T_{temp} \cup$ split($T[t_k, t_b]$)
end if
return T_{temp}

Algorithm 3.1 – The Split Algorithm

merge(T) – merge two adjacent segments on time series T

while *true* **do**
  $\varepsilon_{min} = \infty$
  repeat
    $i = 0$
    $\varepsilon_i = \sum_{j=t_i'}^{t_{i+2}'} (p_i - \widehat{p_i})^2$
    if $\varepsilon_{min} > \varepsilon_i$ **then**
      $\varepsilon_{min} = \varepsilon_i$
      $k = i + 1$
    end if
  until end of time series
  if t-test.accept(ε_{min}) **then**
    drop(t_k, p_k)
  else
    break
  end if
end while
return T

Algorithm 3.2 – The Merge Algorithm

3.3. Support Vector Machines

Techniques in machine learning generally are classified into two main divisions: supervised and non-supervised learning. Support Vector Machines (SVMs) are classified under the former. This means that the process by which SVMs solve tasks is by first undergoing a training phase in which input data (referred to as a set of examples or instances) are used to learn a model that can then be used to classify new instances into a category of a set of categories. SVMs typically solve binary classification[6] problems.

SVMs work by projecting and mapping (by using a kernel function) training data instances into a high-dimensional feature space, in order that the maximum gap between the two categories is created. Hence, new instances can be classified by side of the gap the point falls on. *Gap* is a vague term and is difficult to define. Instead, SVMs work by constructing a hyperplane that separates the two categories – this is referred to as the maximum margin hyperplane. The maximum margin hyperplane is sought as theoretically, such an hyperplane should lead to the least generalisation error.

A hyperplane which defines the decision boundary of the classes is in turn defined in terms of a set of points whose set of points x satisfy the equation.

$$w^T \cdot \phi(x) + b = 0$$

where w^T is the weight vector of the hyperplane, $\phi(x)$ is the kernel method that maps the data points to feature space and b is the bias.

For linearly separable data, two hyperplanes can be defined which completely separate the data such that there are no points between them. The area in space that is bordered by the two hyperplanes is called a margin. The two hyperplanes are defined by the following equations:

$$\begin{aligned} H_1 &: \quad w^T \cdot \phi(x) + b_0 = +1 \\ H_2 &: \quad w^T \cdot \phi(x) + b_0 = -1 \end{aligned}$$

In order to define H_1 and H_2, vectors must be selected which just touch the margins – these vectors are referred to as support vectors as these are the only data instances required to represent the model. The margin can also be defined more formally in terms of d_1 and d_2 which are the shortest distance between the maximum margin hyperplane H_0 and the closest positive support vector and the shortest distance between the H_0 and the closest negative support vector, respectively.

In figure 3.1., $\frac{b}{||w||}$ is defined as the offset between the maximum margin hyperplane and the origin. We can thus say that $d = d_1 + d_2$ is the width between H_1 and H_2. Given that the distance between H_1 and H_0 is $\frac{|w^T \cdot \phi(x)|}{||w||} = \frac{1}{||w||}$, the total distance d is then $\frac{2}{||w||}$. We therefore need to minimise $||w||$ in order to maximise the margin, d. This problem can this be formulated mathematically as:

---

[6] Binary classification involves the training of a model to differentiation between only two classes.

$$y_i(w^T \cdot \phi(x_i) + b) \geq 1 = \begin{cases} w^T \cdot \phi(x_i) + b \geq +1 \; when \; y_i = +1 \\ w^T \cdot \phi(x_i) + b \leq -1 \; when \; y_i = -1 \end{cases} \tag{3.9}$$

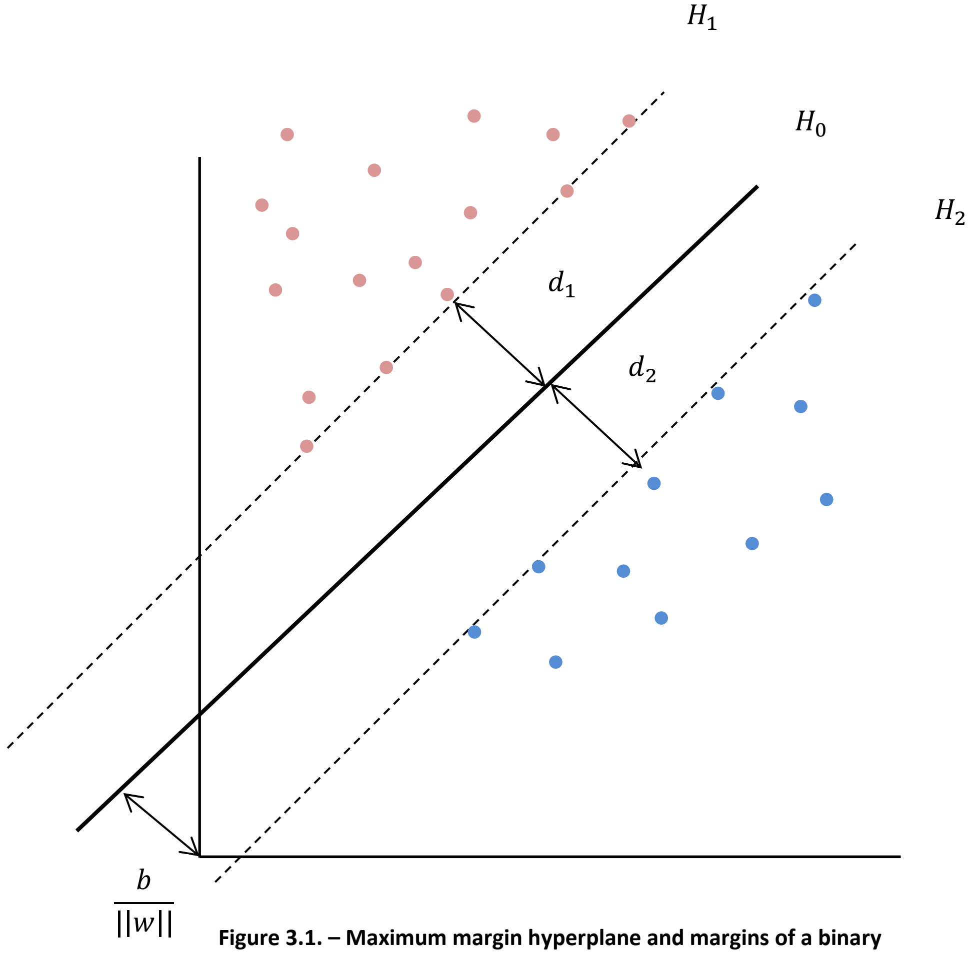

Figure 3.1. – Maximum margin hyperplane and margins of a binary classification SVM-based model

where x_i is an example, y_i is the target class. Minimising $||w||$ can be reduced to $\frac{1}{2}||w||^2$

Having formulated the linearly separable case, we note that not all SVM-applicable problems are linearly-separable; in fact, most of them are not. For this, we introduce a slack variable $\xi_n \geq 0$ which acts as a penalty term for misclassified examples. ξ_n is thus formulated mathematically as:

$$\xi_n = \begin{cases} 0, & \text{if correctly classified} \\ |t_i - y_i(\phi(x_i) \cdot w^T + b)|, & \text{otherwise} \end{cases} \tag{3.10}$$

Hence, $y_i(w^T \cdot \phi(x_i) + b) \geq 1$ can be rewritten as $y_i(w^T \cdot \phi(x_i) + b) \geq 1 - \xi_n$. This new formulation is referred to as a soft margin and the modified optimisation goal is:

$$arg \max_{\xi_n, w} c \sum_{n=1}^{N} \xi_n + \frac{1}{2}||w||^2 \tag{3.11}$$

Often, we find that there are more than a single category to classify and there are two main approaches to multiclass classification problems: one-versus-one and one-versus-rest. One-versus-one attempts to differentiate between each two pairs of class while one-versus-many differentiates between a single class and the other classes.

We conclude our discussion of support vector machines with the kernel function, ϕ. The resulting feature space is heavily dependent on the exact function mapping and there are a few popular kernel functions, which are simply listed here.

a. Linear kernel is defined as $k(x, x') = x^T x'$
b. Polynomial kernel is defined as $k(x, x') = (x^T x' + c)^d$ where c is a constant which accounts for the influence of higher-order terms versus the lower-order terms d is the degree
c. Radial basis function kernel $k(x, x') = \exp\left(-\gamma \left\| x - x' \right\|^2\right)$ where γ is an hyper-parameter referred to as the kernel bandwidth.
d. Gaussian kernel: $k(x, x') = \exp(-\frac{\left\| x - x' \right\|^2}{2\sigma^2})$

3.4. Hidden Markov Models(HMM)

Hidden Markov Models (written as $\boldsymbol{\theta}$)are simply a set of parameters which explain a pattern for a known class or category. HMMs can be used to classify a test-pattern for which it has the highest posterior probability. Sequences can be considered as a series of states $\omega(t)$, written as $\boldsymbol{\omega^T} = [\omega(1), \omega(2), \dots, \omega(T)]$. There are no restrictions on the number of states to be visited or the number of times a state can be visited. In order to define any sequence, transition probabilities –the probability of ω_j at $t+1$ given ω_i at t - must be evaluated. This is formulated mathematically as:

$$a_{ij} = P\left(\omega_j(t+1) \middle| \omega_i(t)\right) \tag{3.12}$$

At each step in the sequence, the state $\omega(t)$ emits a symbol $v(t)$. Symbols are otherwise referred to as observations as they are visible. Hence, we might have a sequence of observations $\boldsymbol{V^T} = [v(1), v(2), \dots, v(T)]$. Given the states and observation, we can then make another deduction: the probability of observing a certain symbol given the state at time t:

$$b_{jk} = P(v_k(t) | \omega_j(t)) \tag{3.13}$$

b_{jk} is thus referred to as the emission probabilities. At each time step, a transition must occur and a symbol must be emitted, resulting the redundant formulations:

$$\begin{aligned} &\textstyle\sum_j a_{ij} = 1 \text{ for all } i \\ &\textstyle\sum_k b_{jk} = 1 \text{ for all } j \end{aligned} \tag{3.14}$$

There are three main problems addressed with the Hidden Markov Model: The Evaluation problem, the decoding problem and the learning problem.

3.4.1. Evaluation

Given an HMM, the transition and emission probabilities, evaluate the probability of a sequence V^T being generated by the model:

$$P(V^T) = \sum_{r=1}^{r_{max}} P(V^T | \omega_r^T)\, P(\omega_r^T) \tag{3.15}$$

The problem can be solved by sum of all r_{max} possible sequences of the conditional probability of the transitions multiplied by the emission probability the sequence. However, this is a computational intensive procedure. Instead the forward algorithm (Algorithm 3.3) is used to solve the problem. For the forward algorithm, we define $\alpha_j(t)$, which defines the probability that an HMM is in state ω_j at step t having already generated the first t elements of the sequence V^T.

$$\alpha_j(t) = \begin{cases} 0 & t = 0 \; and \; j \neq initial \; state \\ 1 & t = 0 \; and \; j \neq initial \; state \\ \left[\sum_i \alpha_j(t-1) a_{ij}\right] b_{jk} v(t) & \text{otherwise} \end{cases} \tag{3.16}$$

The time-inverse algorithm, referred to as the backward algorithm (Algorithm 3.4) can also be used to solve the evaluation problem, and it's necessary to define $\beta_i(T)$:

$$B_i(t) = \begin{cases} 0 & \omega_j(t) \neq \omega_0 \; and \; t = T \\ 1 & \omega_j(t) = \omega_0 \; and \; t = T \\ \sum_i \beta_j(t+1)\, a_{ij} b_{jk} v(t+1) & \text{otherwise} \end{cases} \tag{3.17}$$

forward(T) – calculate $P(V^T)$ recursively

1: **initialise** $t \longleftarrow 0, a_{ij}, b_{jk}$, visible sequence V^T, $\alpha_j(0)$
2: **for** $t \longleftarrow t+1$
3: $\quad \alpha_j(t) \longleftarrow b_{jk} v(t) \sum_1^c \alpha_j(t-1) a_{ij}$
4: **until** $t = T$
5: **return** $P(V^T) \leftarrow \alpha_0(T)$ for the final state
6: **end**

Algorithm 3.3 – the forward algorithm

backward(T) – calculate $P(V^T)$ recursively

1: **initialise** $\beta_j(T) \longleftarrow T, a_{ij}, b_{jk}$, visible sequence V^T
2: **for** $t \longleftarrow t+1$
3: $\quad \beta_i(T) \longleftarrow \sum_{j=1}^c \beta_j(t+1) a_{ij}\, b_{jk} v(t+1)$
4: **until** $t = 1$
5: **return** $P(V^T) \leftarrow \beta_i(0)$ for the known initial state
6: **end**

Algorithm 3.4 – the backward algorithm

3.4.2. Decoding

Suppose we have a observations sequence V^T and a model $\boldsymbol{\theta}$**,** determine the most likely sequence of hidden states that generated the observations. The Viterbi algorithm is used to solve this problem (Algorithm 3.5.)

decoding(V^T) – determine the most likely ω^T

begin **initialise** Path $\longleftarrow \{\}. t \longleftarrow 0$
for $t \longleftarrow t+1$
    $j \longleftarrow j+1$
    for $j \longleftarrow j+1$
        $\alpha_j(t) \longleftarrow b_{jk} v(t) \sum_1^c \alpha_j(t-1) a_{ij}$
    until $j = c$
    $j' = \arg\max_j \alpha_j(t)$
    Append $\omega_{j'}$ to Path
until $t = T$
return Path
end

Algorithm 3.5 – Viterbi Algorithm

3.4.3. Learning

Given the number of states and the number of visible states and a set of training observations, determine the emission and transition probabilities. Starting with estimates of a_{ij} and b_{jk}, we can improve our estimates by first calculating the probability of transition from $\omega_i(t)$ to $\omega_j(t+1)$ given by $\gamma_{ij}(t)$:

$$\gamma_{ij}(t) = \frac{\alpha_i(t) a_{ij} b_{jk} \beta_j(T)}{P(V^T \mid \boldsymbol{\theta})} \tag{3.18}$$

where $P(V^T \mid \boldsymbol{\theta})$ defines the probability that the model $\boldsymbol{\theta}$ generated V^T. The improved estimates $\hat{a}_{ij}$ and $\hat{b}_{jk}$ can then be used in the forward-backward algorithm (also known as the Baum-welch algorithm):

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)} \tag{3.19}$$

and:

$$\hat{b}_{jk} = \frac{\sum^{T}{}_{\substack{t-1 \\ v(t)=v_k}} \sum_l \gamma_{il}(t)}{\sum_{t=1}^{T} \sum_l \gamma_{il}(t)} \tag{3.20}$$

Forward-backward(V^T) – determine $\hat{a}_{ij}$ and $\hat{b}_{jk}$

begin **initialise** $\hat{a}_{ij}, \hat{b}_{jk}$, training sequence V^T, convergence criterion $\theta, z \longleftarrow 0$
do $z \longleftarrow z+1$
    compute $\hat{a}(z)$ from $\hat{a}(z-1)$ and $\hat{b}(z-1)$ by equation 3.19
    compute $\hat{b}(z)$ from $\hat{a}(z-1)$ and $\hat{b}(z-1)$ by equation 3.20
    $a_{ij} \longleftarrow \hat{a}_{ij}(z)$
    $b_{jk} \longleftarrow \hat{b}_{jk}(z)$
until $\max_{i,j,k}[a_{ij}(z) - a_{ij}(z-1), b_{jk}(z) - b_{jk}(z-1)] < \theta$
return $a_{ij} \longleftarrow a_{ij}(z); b_{jk} \longleftarrow b_{jk}(z)$
end

Algorithm 3.6 – the forward-backward algorithm

3.5. Hybrid HMM-SVM Model

The variation in hidden markov models comes in the form of which the emission probabilities are calculated; hence, HMMs can be based on the Poisson distributions, Gaussian distributions or Gaussian mixture models and in this case SVMs. For continuous data, Gaussian mixture models are typically the preferred method for deriving the emission probabilities; however, they are often criticised for their poor discriminatory capabilities. Conversely, SVMs are popular for their great discriminatory capabilities.

SVMs do not directly output probabilities, instead they output the measured distance between the example data and the hyperplane: $h(\boldsymbol{x})$. In order to create a link between the posterior probability of a instance x having a class y: $p(y = +1|x)$ and the result of the SVM, we need to utilise Platt's (1999) proposal for generating pseudo-probabilities by the use of a sigmoid function:

$$p(y = +1|f) = \frac{1}{1 + \exp(Af + B)} \tag{3.21}$$

where A and $B$are derived by using maximum likelihood estimation fron a training set. And f is the unthresholded output of the SVM defined by:

$$f(\boldsymbol{x}) = h(\boldsymbol{x}) + b \tag{3.22}$$

In the binary case, we can thus compute the posterior conditional probabilities $p(c_1|\boldsymbol{x})$ and $p(c_2|\boldsymbol{x})$ of classes c_1 and c_2 given the symbol . Finally using Bayes' rule, we can compute the emission probabilities from the outputs of the SVM using the prior probability $p(c)$ calculated from the frequency of the class in the training data:

$$p(x|c) \propto \frac{p(c|\boldsymbol{x})}{p(c)} \tag{3.23}$$

3.6. Technical Indicators: A Description.

Our selection of technical features is based on the work of Kim and Han (2000) who in turn, base their selection on the review of financial experts. The list of technical features and the formulas for how they are derived is provided in figure 3.2.

The Stochastic Oscillator $\%K$ developed by George Lane (1998), compares the closing price with the price range over the previous period. $\%K$'s sensitivity can be altered by changing the length of the period or by introducing a momentum. The stochastic oscillator is therefore used to identify bullish[7] or bearish[8] deviations in a time series – which can then be used to predict reversals in trend line directions. The moving average of the stochastic $\%K$ (referred to as the stochastic $\%D$) is used for less sensitive comparisons to the market. For even less sensitivity, the Stochastic Slow $\%D$ is used.

The momentum is used to determine by how much the closing price has changed over a period. Williams $\%R$ is an indicator of the price momentum which serves to compare the close price

[7] Bullish market periods are periods in a financial time series in which prices are on the rise.
[8] Bearish market periods are periods in a financial time series in which prices are on the fall.

with the highest high price over a period. Correcting for inversion, the Williams $\%R$ multiply the value by -100, which results in values within the range of 0 to -100. $\%R$ reflects whether a stock has been oversold (%R values of less than -80)or overbought (%R values of less than -20).

Another technical indicator that can be used to define overbought or oversold stock is the Rate of Change (ROC). The ROC is defined by calculating the difference in the current close price and the close price n days ago and is typically used by analysing for positive and negative divergence in the ROC values.

The Accumulation/Distribution (A/D) oscillator measures whether investors are buying or selling a stock and is also a measure of momentum. The A/D Oscillator works by associating changes in price which is in other words, identifying trends.

The 5 and 10-day disparity measures give a value to the difference between the current closing price and the moving average over the past 5 or 10 days, respectively.

The price oscillator measures the difference between two moving averages. A positive value for this technical indicator reflects an upward price trend and the reverse the case for a negative value. In addition, the price oscillator often acts as a normaliser so that the price oscillator values of one stock can be compared with that of another stock.

The variation between the close price and the statistical mean of a stock is referred to as the commodity channel index (CCI). The CCI was introduced by Donald Lambert (1980) to identify new trends and serves as a warning for extreme conditions in the commodities market but is also applied to the stock market. A high CCI indicates that prices are unusually high above their average while a low CCI price reflects that stock prices are extremely low below their average. As with some other indicators, the CCI can be utilised in identifying overbought or oversold stocks.

The relative stock index (RSI) developed by Welles Wilder measures the speed at which price changes and hence is another momentum indicator. RSI values fall within the range of 0 to 100 and also indicate the overbought (RSI above 70) or oversold (below 30) status of a stock.

Technical Indicator	Formula
Stochastic %K	$\frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \times 100$
Stochastic %D	$\sum_{i=0}^{n-1} \frac{\%K_{t-1}}{n}$
Slow Stochastic %D	$\sum_{i=0}^{n-1} \frac{\%D_{t-1}}{n}$
Momentum	$C_t - C_{t-4}$
Rate of Change	$\frac{C_t}{C_{t-n}} \times 100$
William's %R	$\frac{HH_n - C_t}{HH_n - LL_n}$
A/D Oscillator	$\frac{H_t - C_{t-1}}{H_t - L_t}$
Disparity5	$\frac{C_t}{MA_5} \times 100$
Disparity10	$\frac{C_t}{MA_{10}} \times 100$
Price Oscillator	$\frac{MA_5 - MA_{10}}{MA_5}$
Commodity Channel Index	$\frac{M_t - SM_t}{0.015 D_t}$
Relative Strength Index	$100 - \frac{100}{1 + \frac{\sum_{i=0}^{n-1} \frac{Up_{t-i}}{n}}{\sum_{i=0}^{n-1} \frac{Dw_{i-1}}{n}}}$

where C_t is the close price, HH_t is the highest high and LL_t is the lowest low for the last t days, H_t is the high price at time t, L_t is the low price at time t, MA_5 is the 5 day moving average, MA_{10} is the 10 day moving average, $M_t = \frac{H_t + L_t + C_t}{3}$, $SM_t = \frac{\sum_{i=1}^{n} M_{t-i+1}}{n}$, $D_t = \frac{\sum_{i=1}^{n} |M_{t-i+1} - SM_t|}{100}$, Up_t the upward price change at time t and Dw_t is the downward price change at time t.

Figure 3.2 – Technical Indicators and their formulas

4. Scientific Method

4.1. Method Overview

The proposed system is comprised of two main parts: sentiment-based news classification and price prediction. Any developed system cannot completely rely on news as news is sporadic and there's no guarantee that news regarding a certain entity will be released for every single trading day and thus, there needs to be a means by which prediction can still take place. This comes in the form of technical data (section 3.6.). Hence, there are two phases of classification – the classification of pure news articles and the classification of technical data. Figure 4.1 shows the system overview, ignoring news labelling.

4.2. Sentiment Classification

As emphasised in previous sections, the first task to be performed is the classification of news articles. As with classification of other types of data, the following steps need to be performed: data acquisition, data labelling, data pre-processing, data analysis and then finally classification. The range of human sentiment is very wide and includes sentiment such as happiness, sadness, calmness, anger and anxiousness. This range is much too wide for the application at hand. In fact, we have taken a much simpler approach and simply classified the sentiment of the news articles into three categories: happy, sad and ambivalent.

Unlike sentiment classification for domains such as films, cars or music, happiness, sadness and ambivalent news articles may not bear much information about the progress of the entity. While the sentiment of the article is what we aim to extract, a cursory look at any news article that bears financial information will show that news articles aren't very sentimental. This indicates that classification based simply on human sentiment while might be accurate might not be as successful given that we aim to predict the stock market. Therefore, to supplement classification based on sentiment, we also classify based on the progression of the company. This means that the articles get classified into an additional set of categories (positive, negative and neutral).The aim with the progress classification is that articles get classified based on what the news article's evaluator expects as regards to whether the entity's stock price will go up, down or simply stay the same as a result of the article.

This new direction of evaluation however raises the question of how to gauge the effect of a news article. For illustrative purposes, if the price of a specific stock has been on the rise for the past three days and then a news article is released and it's classified as "positive", how do we factor that in? Does it simply not matter as we have a direction of progress or could we instead watch for the rate of the change of the stock price? We aim to answer some of these questions in the following sections and the over the next chapter.

4.2.1. Data Acquisition

4.2.1.1. News Articles Acquisition

It's clear that the very first task to be performed is the acquisition of news articles whether labelled or non-labelled. Although a fair bit of work have been done using this particular approach to stock price prediction, we were unable to find any publicly available datasets that fell in line with the purposes of this dissertation. Hence, a dataset was generated from online news sources.

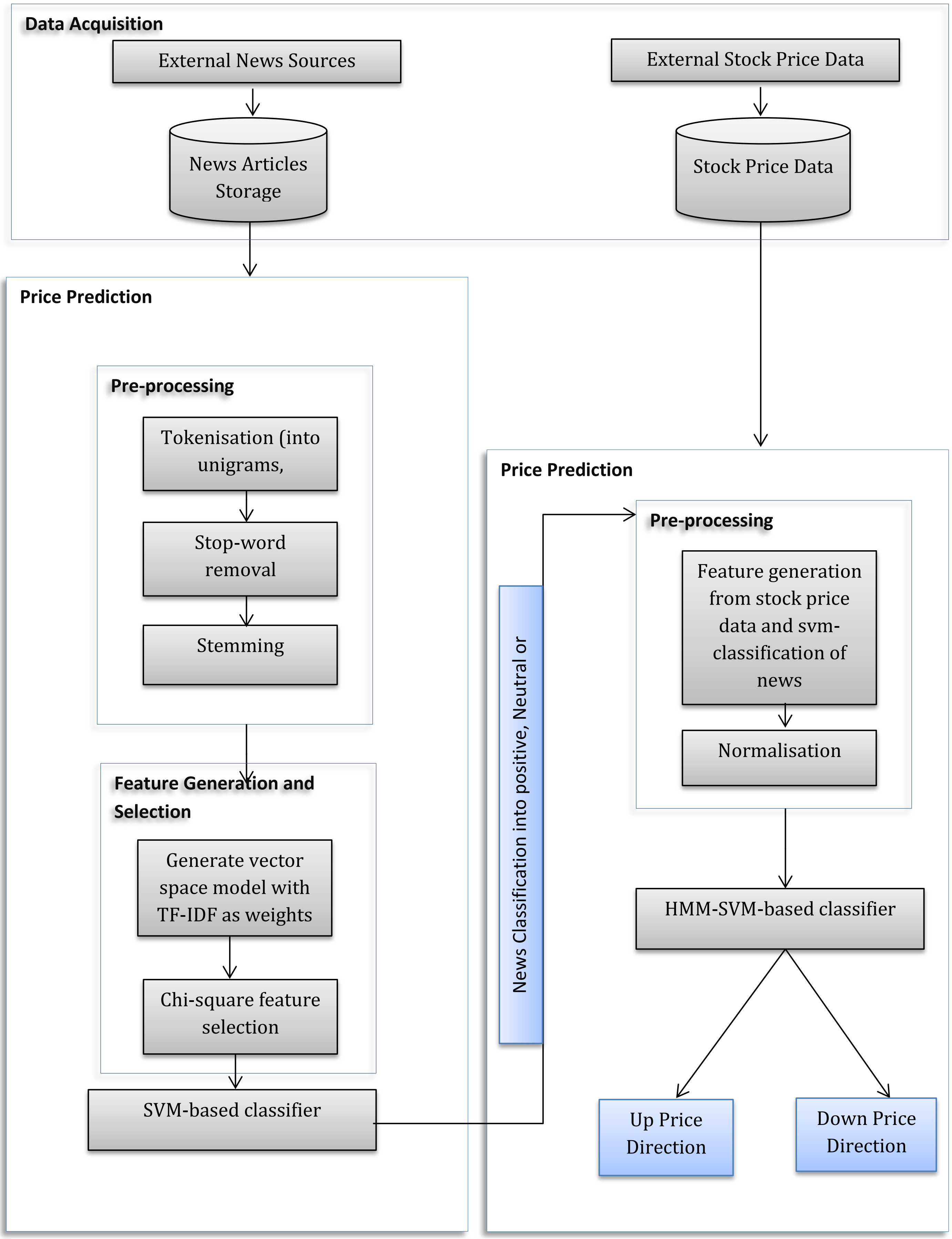

Figure 4.1 - Process Overview

Although selecting news sources seems like a trivial task, it requires careful consideration as the news sources has to be able to satisfy the following requirements:

i. Has to be popularly read, especially by traders. This is particularly important because a high level of trust needs to be placed in the news source, enough to determine that significant changes in stock price trend will be reflected in the news articles.
ii. The news sources should have decent coverage of news– to ensure that we gather as much data as possible.

Given these requirements, investors on online forums (as well as individuals with knowledge of finance) were asked which news sources were read and the following sources were given: Reuters, Bloomberg, Financial Times, Market Watch, Yahoo Finance.

The next step is to scrap selected websites (Bloomberg, Reuters) for news articles. We do not discuss the exact process of scraping websites as it's not relevant to this project. However, scraping can involve interesting problems such as logging in to websites via a program (in this case, python) and extracting data.

Prior to the scraping of news articles, we must first determine what it is we hope to find – in this case, we want to scrap enough news articles in order to perform classification on the news articles. Hence the gathering of news articles has to be targeted so that we have enough data for each of the companies we aim to classify

Finally, the news data is extracted and put in the following xml format:

```xml
<?xml version='1.0' encoding='us-ascii'?>
<news>
        <entry author=''…'' datetime=''…'' url=''…''>
                <headline>
                        …
                            </headline>
                <body>
                        …
                </body>
        </entry>
</news>
```

Figure 4.2 – XML Format for Scrapped News Articles

4.2.1.2. Stock Price Data Acquisition

The acquisition of the numerical stock data simply involves selecting the companies which are of interest and extracting the data from Yahoo Finance. In this case, the API (ystockquote) was used to extract the data for relevant companies. The data collected for each company includes the following: Date, Adjusted Close, Close, High, Low, Open, and Volume.

4.2.2. Data Labelling

At the start of the project, the intention was to crowd source the labelling of the articles. This would be done by asking individuals with knowledge of economics and finance to evaluate the news articles. This involved uploading the corpus to a website for easier classification. (sentimentanalysis.bolanleonifade.me). However, the rate at which the articles were getting classified was very slow so the alternative approach taken was to use prior knowledge (of the current author) to label the article. This of course meant that experiments might suffer due to lack of enough financial knowledge. Therefore, in order to provide a baseline or at the very least, a means of evaluating the manually labelled data, a set of automatically labelled data was created as well.

4.2.2.1. Manual Labelling

Manual labelling of data is simply reading each news article and labelling them by hand. The evaluators are asked to estimate the company's progression based on the news article. Their estimates can fall into the following categories: (up, down, neutral). The evaluators are also asked to provide the sentiment of the article (happy, sad, neutral). From henceforth, for clarity purposes, we shall refer to the former as "progress sentiment" and the latter as "feeling sentiment"

One might think that there is a perfect correlation between the two sets of categories. However, there can be differences between the two. For illustrative purposes, we will examine a couple of cases in which there are differences:

The headline "Exxon Mobil reports Fire, oil spill at Nigerian terminal", evokes a feeling of sadness but due to the established nature of Exxon, it's unlikely that this event is going to lead to a massive dent in the stock price, we give the article a progress sentiment of neutral (because the article doesn't go on to indicate that Exxon will suffer from this incident). Another article that wouldn't be expected to change the stock price much is "JP Morgan employee falls to death from building roof in Hong Kong". It's however clear that the article is "sad", but the effect on JP Morgan's progress would virtually be nil.

If the previous two examples give the impression that from headlines, we can always tell the feeling sentiment of an article, it would be wrong. In fact, a seemingly neutral headline such as "Coca-cola names Walter Finance Chief as Fayard Retires" goes on to discuss the recent struggles of coca-cola, therefore giving it a feeling sentiment of sad and a progress sentiment of neutral. In the same strain, we discovered articles can both be up for progress sentiment and feeling sentiment; this would be the case for articles that discuss an entity's growing business.

4.2.2.2. Automatic Labelling

In order to perform automatic labelling of news articles, we need to generate projected trends, this gives us an idea of the overall outlook of the stock price – that is, for example, we can safely say that the overall projected trend of the stock price is an upwards movement if the price over a period of time has changed positively, ignoring every minor dip in the price trend. We can perform automatic labelling by using piecewise linear approximation. This allows us to align news articles with the projected stock price and simply labelling the articles based on the projected stock price.

There are obvious inaccuracies that can occur from the use of such a method – in fact, as shown in the literature review, classification based purely on price differences, tends not to be very accurate – this is because one cannot say for sure that all articles released during periods of overall upwards positive price movement are positive and vice versa. However, we are operating under the assumption that news articles strongly reflect the direction of movement of the stock market. We expect the price to move up when news articles discuss increases in sales, innovation, positive restructuring and we expect the price to move downwards when news articles discuss fines, bankruptcy, litigation, sanctions etc.

Automatic labelling, therefore provides us with a baseline. The higher the similarity between the results of automatic labelling and that of manual labelling, the more *trust*, we can place in the results of manual labelling.

4.2.2.3. Evaluating Labelled News Articles

Sentiment labelling generated via automatic categorisation is a reflection of the price movements, not a reflection of the articles themselves. However, since the articles themselves are manually labelled to reflect precisely the sentiment which they carry, we can conclude that if there exists a high similarity the results of manual labelling and the results of automatic labelling, then we can say confidently that the labelled articles can led to positive results in later classification.

We note that news articles labelled automatically cannot be classified based on feeling sentiment. Feeling sentiment by definition requires an evaluator to label articles based on what feelings are evoked by reading the article. However, since the method by which we automatically label stock data is based on the progression of the stock price, we can automatically label articles based on progress sentiment.

How therefore do we evaluate the results of labelling? An easy method of doing is by calculating the Pearson's correlation coefficient. We comprehensively discuss the results of the calculations and the other considerations (specific to calculating the correlation) when discussing the evaluations in chapter 5). We finish off this section by pointing out that the more similar the correlation values are between the projected trend line (generated via piecewise linear approximation) and the actual stock price trend line, the higher the similarity is between the two trends.

4.2.3. Data Pre-processing

Retrieved news articles are in HTML format. The news articles therefore need to be converted into plain text, tokenised (into unigrams and bigrams), stemmed and have stop words removed (discussed in chapter 3) before classification can take place.

4.2.4. Document Representation

After the completion of all pre-processing steps, the documents are now ready to be transformed into vectors. The library Scikit-learn provides the TfidfVectorizer that converts the news articles into aTF-IDF-weighted document-term matrix. TF-IDF has been discussed in (chapter 3).

4.2.5. Feature Selection or Reduction

In the literature, the chi-squared method for feature selection and the SVD method are popular and but we decided to use χ^2 as SVD is a very expensive method.

4.2.6. Classification

The final step is to train the SVM to predict the news article. In order to truly evaluate the SVM, cross validation over the data set was performed. 10-fold cross validation ensured we got the performance of the hyper-parameters of the SVM. The results of each fold were evaluated using the following metrics: confusion matrix, recall, precision and f-measure. These results can then be averaged over the number of folds to determine the overall performance of the hyper-parameters.

4.3. Price Prediction

Price prediction is the actual step that combines the results of sentiment classification into a single result. We have introduced all the techniques relevant for price prediction in chapter 3, thus we will dive into the method of prediction.

In order to make price predictions for each individual company, the news data have to be re-split into categories defined by the company the news is relevant to. Thus, we have 14 features for training the SVM-HMM model: 12 technical indicators and 2 news-based features (progress and feeling). For each company, 120 days of news classified by the SVM in section 4.2.6, combined with 120 days of technical data served as the test period while a definable and changeable number of days served as the training period. We say definable as we use a sliding window of training data to train the hybrid SVM-HMM model. This ensures that the model is as sensitive as permissible to the market as possible.

Manual analysis of the piecewise linear approximation-based trend line, gives an indication of how long the trends last for and thus the number of days for the training window. Once determined, we used a fixed training window width. Defining an appropriate window width has significant bearing price prediction; longer window width during times of stability is likely to lead to better results than shorter widths. Correspondingly, a shorter window width during times of rapid swings in the price might lead to better prediction. Keeping this in mind, it has to be noted that the SVM, in order to compute usable probability has to be trained on enough data, hence there is a trade-off between the number of training data that can be used for SVM training and the sensitivity to the market in the short-term. Having defined the window, the SVM is trained with data within the bounds of the window. The SVM can then provide the emission probabilities as explained in section 3.6.

Our definition of the HMM is almost complete as we have the number of states (two, each representing the upwards and downwards movement of the stock price) and the emission probabilities. The transition probability however, is still unknown so we train using the Baum-Welch algorithm for the transition probability. Finally, we can then run the Viterbi algorithm on the past n days of data to predict the sequence of $hidden$ states for each of the past n days. As we are only interested in predicting the next day's price, we simply take the hidden state for the n^{th} day as the prediction of the next day. We formalise the process of making a prediction in algorithm 4.1

price prediction

for each trading day
    select training window w of length m
    using cross-validation and grid-search determine parameters p for SVM
    train SVM for emission probabilities using w, using parameters p
    train HMM for transition probabilities using w
    hidden_state of $n \longleftarrow$ viterbi algorithm using past n days of data
    yield hidden_state of n

Algorithm 4.1 – Price Prediction Algorithm

5. Evaluation and Results

5.1. Method Overview

In this section, we use detail the results of each of the main activities that have been discussed in section 4. The format of this section is the same as that of section 4, this is for easy referencing and comprehension.

5.2. Data Acquisition

5.2.1. News Article Acquisition

We do not use all of the acquired data however because of time constraints – there's no way for a single person to manually label the 12000 articles in the time frame of 4 weeks. Hence, we had to discard a lot of the news articles and aim for classifying a fraction of the news articles (about 2250 articles). There is a need to determine which companies we aim to classify. The companies selected were chosen from the Dow Jones Industrial Average (DJIA) because the companies listed on the index are major American companies which tend to get a lot of attention from the media.

The table below shows the number of articles that were gathered for each company.

Company Name	Number of Articles
Chevron	88
Cocacola	52
Disney	108
Exxon	120
Goldman	731
IBM	118
JP Morgan	613
Microsoft	259
Pfizer	115
Visa	48

Figure 5.1 – Number of articles collected for each company

5.2.2. Stock Data Acquisition

The price daily values were collected for the period from the 1st of January, 2013 to the 30th of September, 2014. Of course, the stock price data is only released for working days so this accounts for only 440 working days.

5.3. Labelling

5.3.1. Manual Labelling

We show the results of manual labelling in this section. Principally, this comes in the form of the Pearson's correlation coefficient carried out between the stock price and the aggregate sentiment. The aggregate sentiment is calculated simply by adding the sentiment for every previous day in the time series. In order to do this, we need to calculate the aggregate sentiment for each working day. This is simply done by adding together all the sentiment values for each article released in the working day.

Supposing therefore that we have 4 articles released on any day x labelled as such: $[-1,-1,0,1]$ (please refer to the appendix – section 7.1 – for how this is derived from the sentiment labels) using progress sentiment. The aggregate sentiment would be given as -1 by adding all the sentiments together. In addition to this, supposing we have any set of 8 days for which the aggregate sentiment for each day is given, we have aggregate sentiments for each day to be the sum of previous sentiments:

$$[4,-2,-1,0,1,2,-1,0] \rightarrow [4,2,1,1,2,3,2,2]$$

We've aggregated the sentiments to better show the rise and fall of the. With this transformed sentiment, we calculate the correlation between the stock price closing values and the sentiment. The table below shows the correlation values between each company's stock data and the aggregated sentiments.

Company Name	Correlation of Progress Sentiment(Actual)	Correlation of Feeling Sentiment(Actual)	Correlation of Progress Sentiment(Projected)	Correlation of Feeling Sentiment(Projected)
Chevron	51.7	51.8	49.7	50.1
Cocacola	32.8	32.6	32.99	32.84
Disney	97.6	97.6	97.10	97.13
Exxon	79.0	78.8	79.7	79.6
Goldman	76.74	77.42	71.98	72.5
IBM	53.2	53.3	51.7	51.8
JP Morgan	84.2	84.4	84.5	84.7
Microsoft	95.8	95.8	95.8	95.8
Pfizer	51.1	51.6	50.9	51.3
Visa	89.14	89.20	88.80	88.86

Figure 5.2 – Table of correlation between sentiment and stock price using manually labelled data

Looking at the table above as well as Figure 5.1, there are several points raised regarding the values shown. We attempt to discuss some of them here but leave others until we show the results of automatic labelling. In addition, we also generate correlation coefficients between the projected trend lines (from piecewise linear approximation) and the sentiment – for purposes of comparison with the results of automatic labelling.

We see that in almost all cases, the values of the correlation projected values are almost always lower than the correlation between the actual stock prices. It is expected that there would be a difference between the correlation between the two pairs of values but assuming that piecewise linear approximation generates trends as accurately as possible, the difference between the correlation values should never be large enough to raise questions. Exxon as we can see has correlation coefficients that are larger in the projected trend line than in the actual trend line

In addition, we can see that although feeling sentiment and progress sentiment are intended to be different measures of labelling articles, they lead to in all cases, similar correlation coefficients. However, we note that in almost all cases, actual progress sentiment has a higher correlation value than projected progress sentiment.

Finally, we address the fact that something must be done for those days in which news articles are not released (this applies as well when calculating the correlation for automatically labelled data). There are three assumptions we can make when classifying based on progress:

1. There is an underlying feeling of positivity – that is, these cases, the stock price is assumed to go up meaning that the progress sentiment is always positive
2. There is an underlying feeling of neutrality – the stock price will stay the same when there's no news.
3. An underlying feeling of neutrality – the stock price will go down when there's no news

While it might seem that an underlying feeling of neutrality is the most appropriate, this is not true. The underlying feeling depends on the company being discussed. Prices of most companies change positively when there's no news hence we assume an underlying feeling of positivity. However, for companies such as IBM, there's an underlying feeling of negativity. This is determined by simply looking at the overall projected trend line. IBM over the course of the time period has shown a gradual decrease in price. This is sentiment is well reflected in articles as IBM over the course of the time period had trouble keeping up with other technology firms who have incorporated cloud computing into the services offered, which IBM had yet to do.

The term *underlying feeling* is vague and thus, explaining what is meant by it is important. Prices rise when traders feel that a stock is undervalued (hence, their demand for it increases), conversely prices fall when traders believe that a stock is overvalued (hence, demand falls). The belief that a stock is overvalued or undervalued changes with events – that are hopefully reported in the news. In the absence of news, we need to still capture this belief or feeling that traders have towards a certain stock. While technical indicators *imply* this feeling, we can explicitly state this feeling as sentiment for those days that news articles aren't released.

News articles aren't released solely during weekdays; however stock markets are closed over the weekend. This doesn't mean that trading does not occur over the weekend – extended hours trading does – however prices changes aren't released for the weekend. Hence, in order to calculate the correlation, we have to eliminate news articles that are released on days that fall on weekends. This of course is detrimental to our calculations but despite this we see that there still is a high correlation between stock prices and news articles.

5.3.2. Automatic Labelling

The first few steps needed to be carried out are similar to the steps carried out for manual labelling – summarily, calculating the aggregate sentiment of the news articles over the time period. As previously mentioned in section 4.2.2.2, we only calculate the progress sentiment of the news articles. The sentiment is then correlated both with the projected trend and the actual price trend. Figure 5.3 - 5.12 show generated projected trend lines as well as the actual stock price movements for the companies.

Using the Exxon trend line as a sample case, we show how news articles are classified based on where they fall in the projected trend line – Figure 5.13

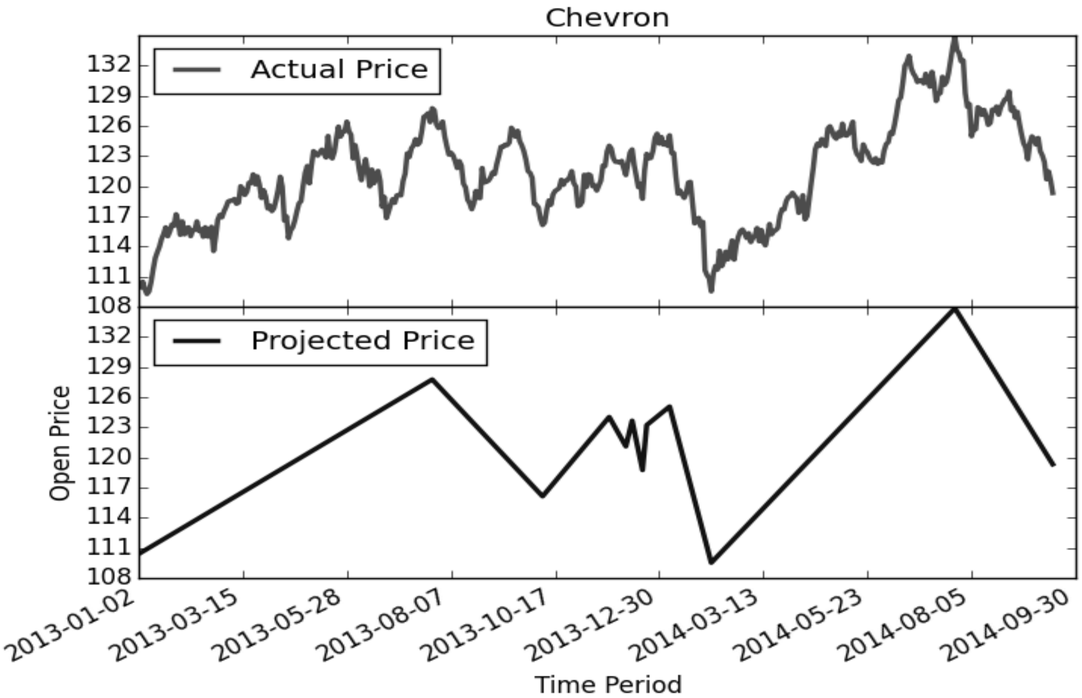

Figure 5.3 - Actual Stock Price and Projected Price of Chevron

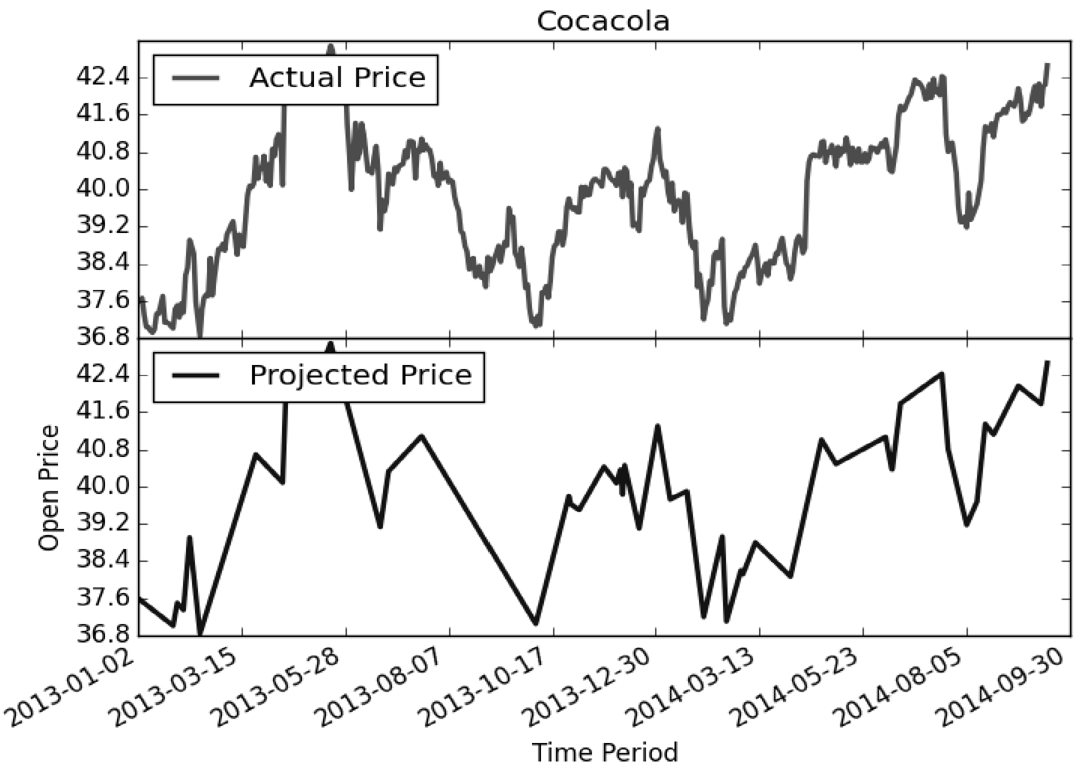

Figure 5.4 - Actual Stock Price and Projected Price of Cocacola

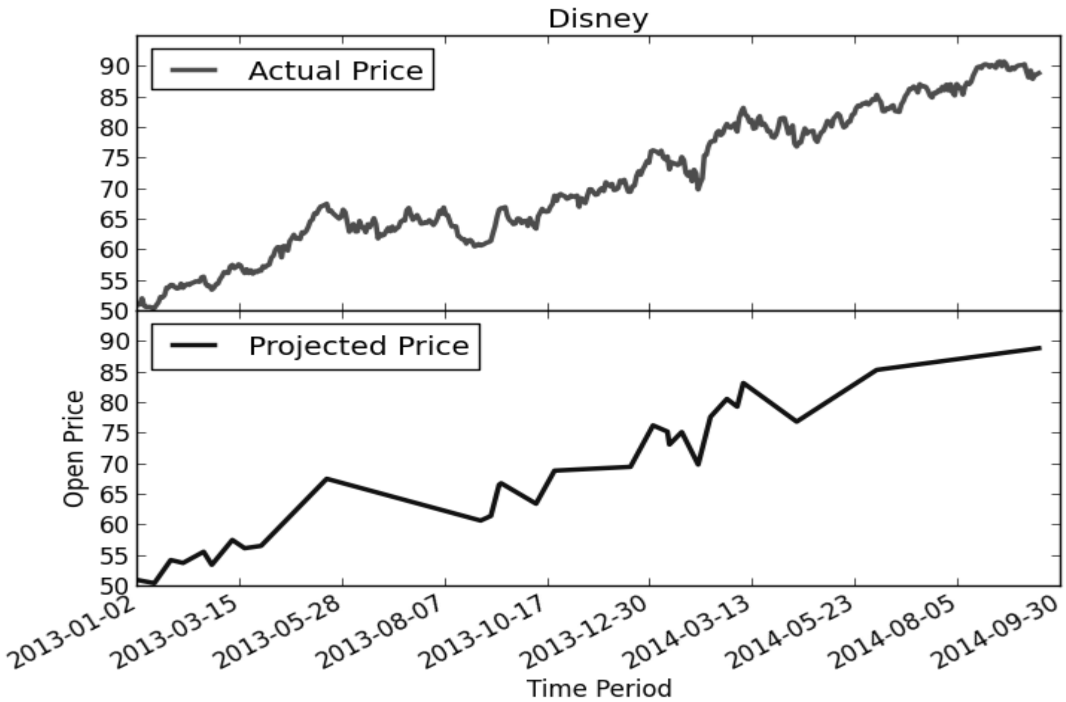

Figure 5.5 – Actual Stock Price and Projected Price of Disney

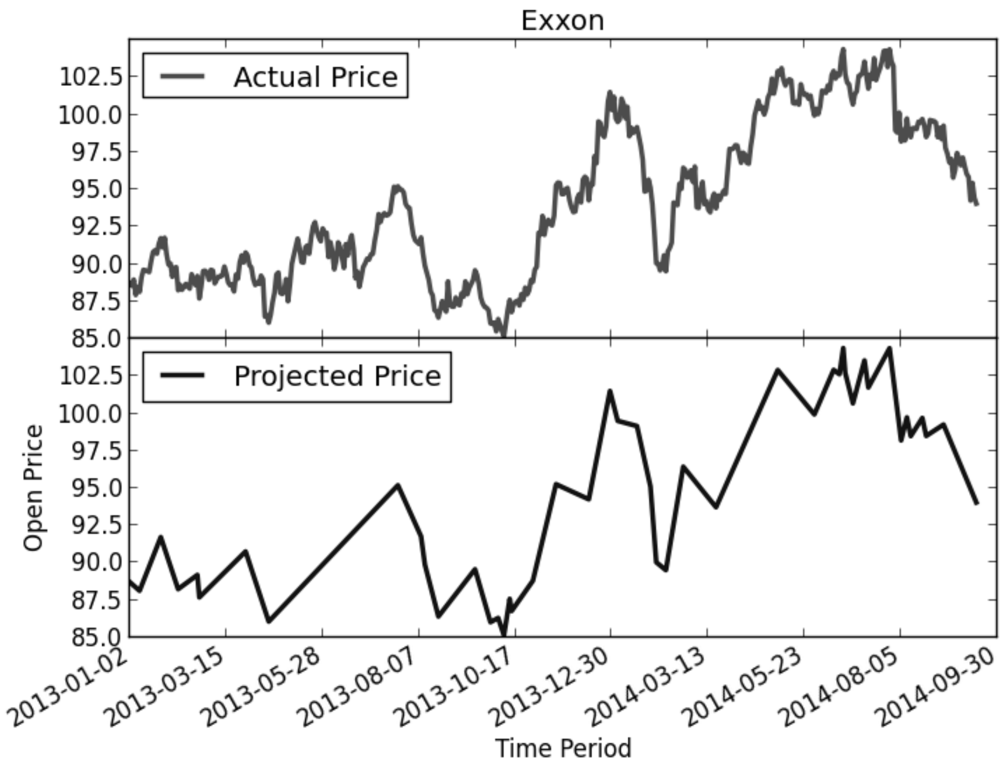

Figure 5.6 – Actual Stock Price and Projected Stock Price of Exxon

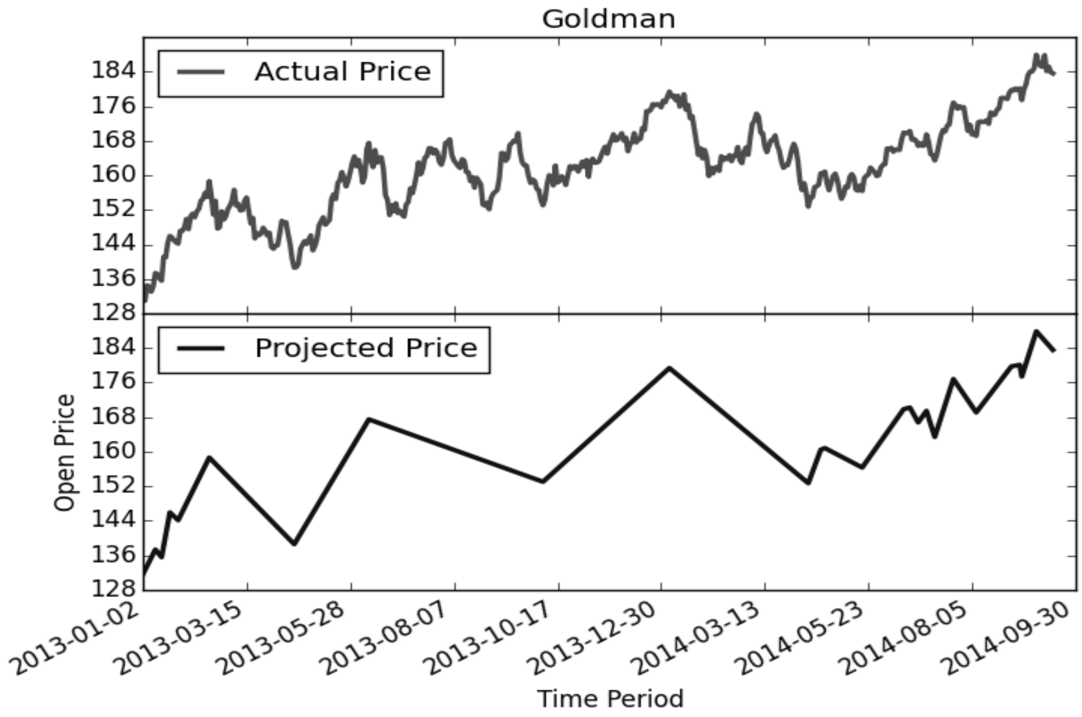

Figure 5.7 - Actual Stock Price and Projected Price of Goldman

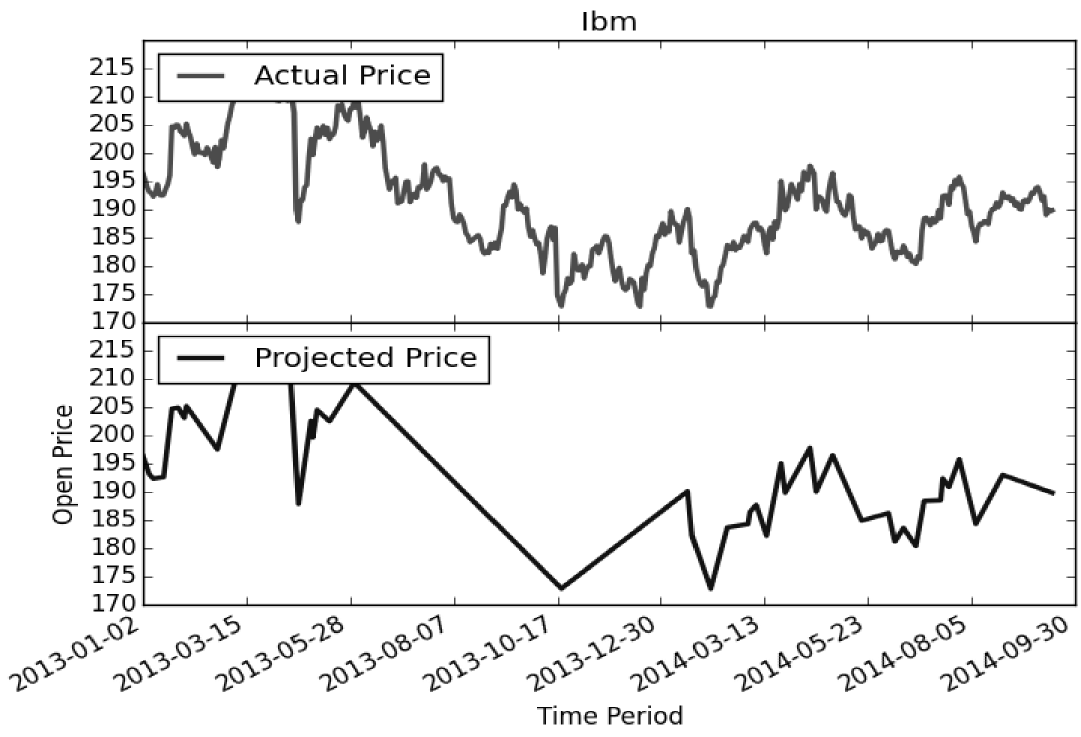

Figure 5.8 - Actual Stock Price and Projected Price of IBM

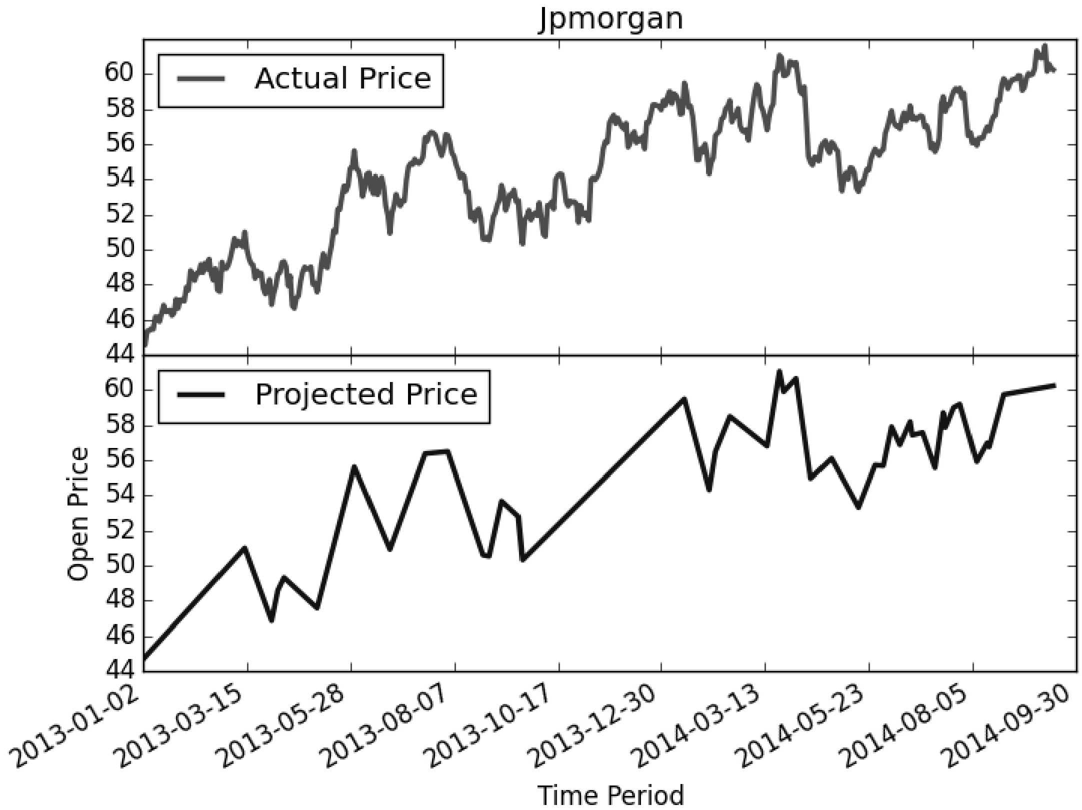

Figure 5.9 - Actual Stock Price and Projected Price of JPMorgan

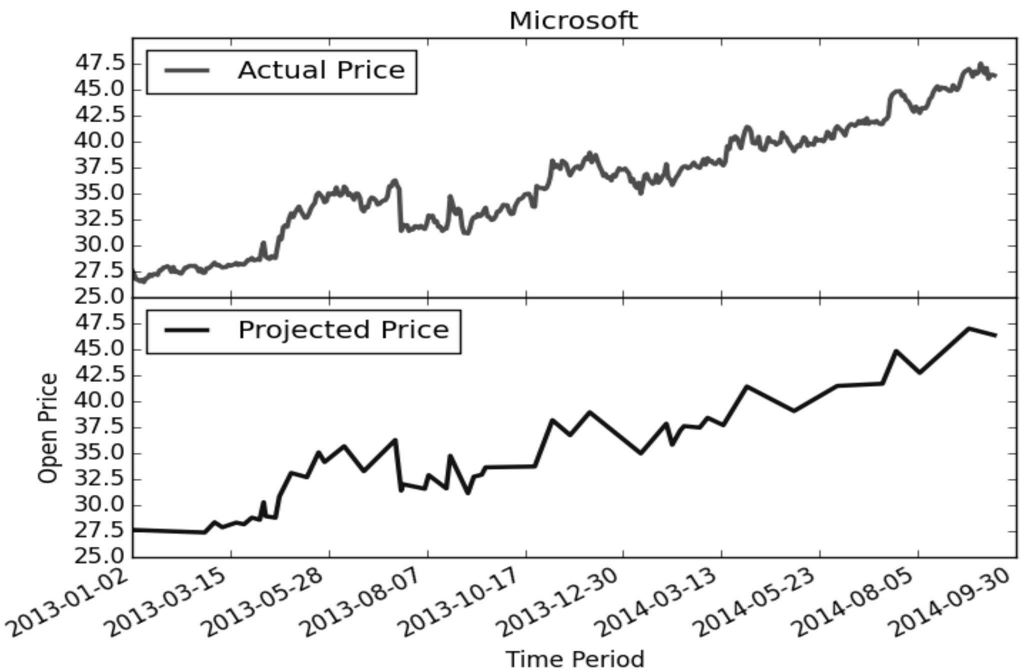

Figure 5.10 - Actual Stock Price and Projected Price of Microsoft

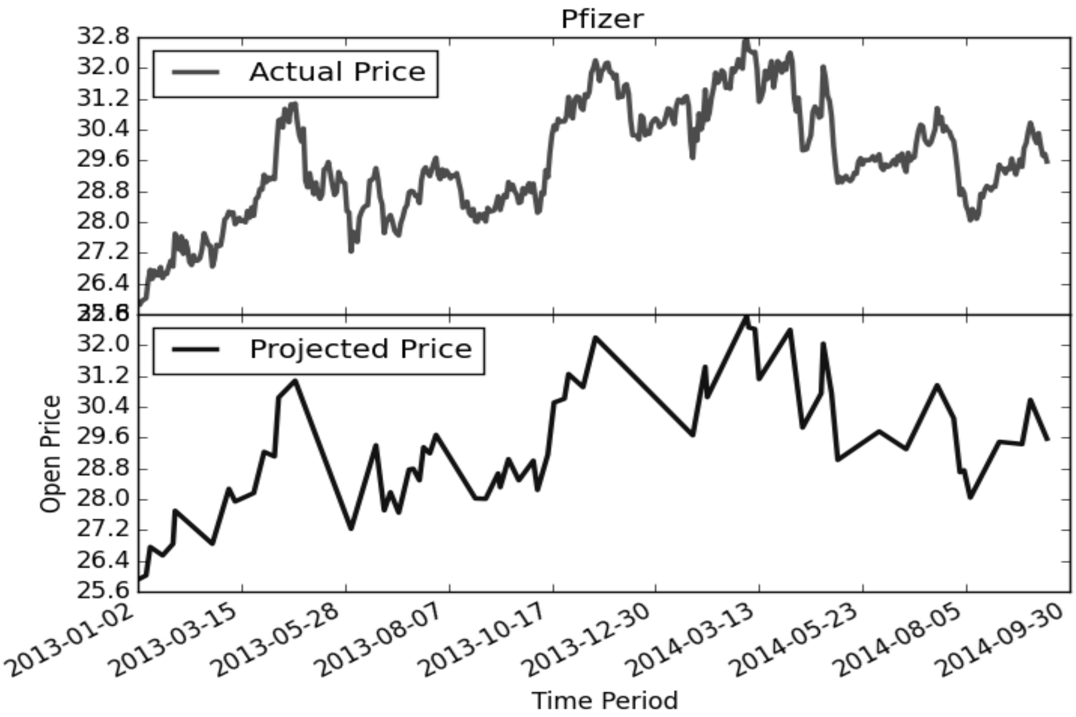

Figure 5.11 - Actual Stock Price and Projected Price of Pfizer

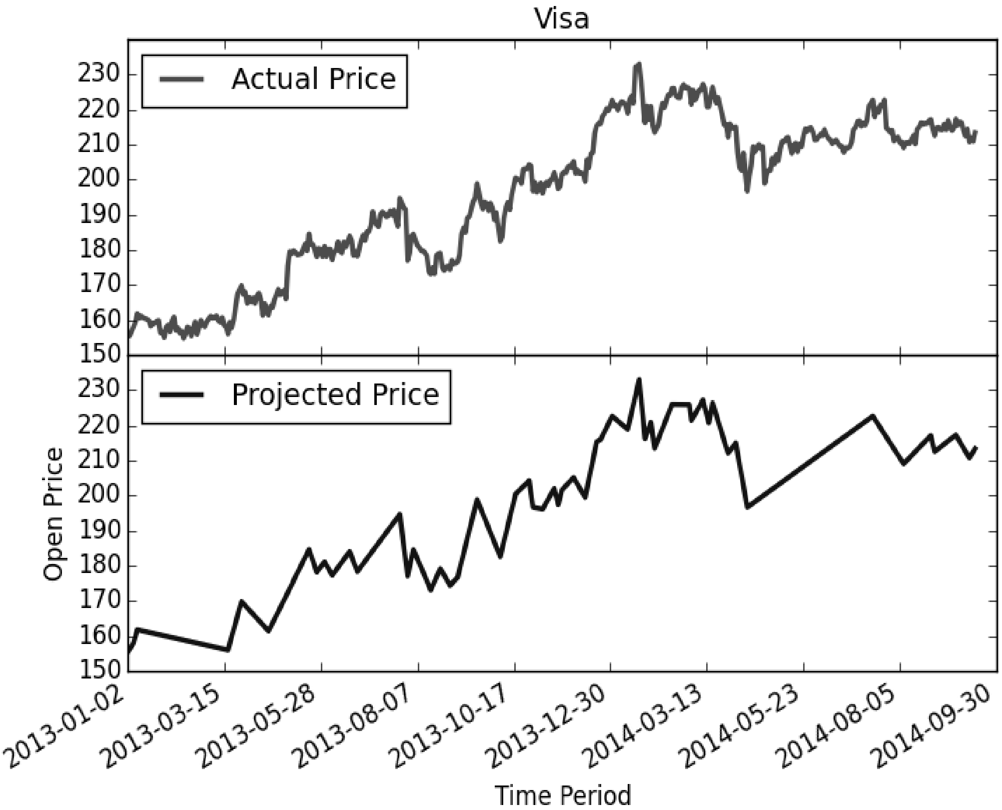

Figure 5.12 - Actual Stock Price and Projected Price of Visa

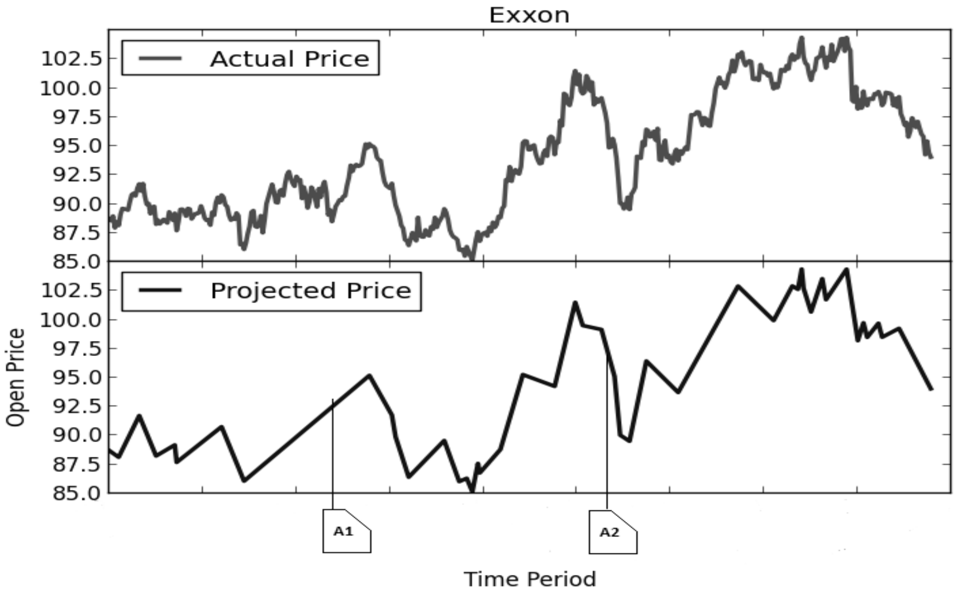

Figure 5.13 – Aligned News Articles with Trends

Using Figure 5.13 as a sample case, we would classify article A1 as "up" or positive while A2 would be classified as "down" or negative. After achieving this step, we can then proceed to calculate the correlation between the automatically calculated prices and the labelling. The table below shows the correlation results.

Company Name	Correlation of Progress Sentiment(Actual)	Correlation of Progress Sentiment(Actual)
Chevron	50.2	51.8
Cocacola	32.1	32.0
Disney	97.0	97.5
Exxon	80.4	79.7
Goldman	50.0	52.60
IBM	52.12	53.41
JP Morgan	85.37	85.14
Microsoft	94.99	94.99
Pfizer	53.08	53.33
Visa	89.27	89.62

Figure 5.14 – Correlation values of the automatically generated data

Looking at the data above and comparing them, we see that the correlation values are quite similar to that of the manual data except for the correlation values of Goldman Sachs which is vastly different from the result of manual classification. The reason for this is that Goldman

Sachs often is the source of the news (for example, Goldman Sachs often advises on buying and selling other companies) and the news is not about Goldman Sachs. This means that automatic labelling is blind to these issues as it labels both news *by* Goldman as well as news *about* Goldman without using any filter. However, when manually labelling, we ensure to label those articles are "neutral" in terms of progress of the entity and the relevant feeling sentiment.

5.3.3. Labelling Discussion

We see that our results aren't perfect – we believe that this is more indicative of the small dataset than a general inability of sentiment to correlate with the stock price.

As this is an exploratory project, the results of labelling is in general very positive – providing a reason to go on with classification of the news articles – preferably using the manually labelled data. The high similarities between the progress coefficient of the automatic and manual data also is a form of validation for the manually labelled data and idea that news articles correlate with the stock price. However, the results also make harder to overlook the issues with automatic labelling.

5.4. Data Pre-processing

Using the method described in section 4.2.3, the dataset was pre-processed. However, there's a decision to be made about which method of tokenisation is best. In our experiments, we performed only 3 types – unigrams, bigrams and combination of both. These types are the most popular in the literature. Unigrams, also known as bag of words are criticised often for not bearing enough information but we see that in all areas, they perform quite well. Bigrams, as we will see also perform comparatively to unigrams. The combination seems to perform the best of all three. In addition, when tokenising, we only select words that have greater than three characters.

The table below shows the number of features that are extracted and the number pre-selected (based on term frequency) before feature extraction or selection.

	Unigram	Bigram	Unigram + Bigram
Initial number of features	26322	310660	336982
Selected amount of features	11000	16000	16000

Figure 5.15 – Initial features and pre-selected features

5.5. Feature Selection or Feature Reduction

During the experimentation phase, we compared the results of SVD with χ^2based feature reduction. The results are, very similar with no distinct advantage provided by SVD (rather, it's disadvantageous as it took up to 2 hours to compute bigrams), we decided to opt for feature selection based on χ^2, hence the classification results in the succeeding section detail the results based on feature selection and not feature reduction simply because the results are virtually the same. Even furthering our decision to use χ^2 is the fact that the computational intensive process of SVD will be even more pronounced with a live system.

In addition to the benefits provided by χ^2, there are also benefits in terms of ability to examine the features selected more closely. In figure 5.9 and 5.10, we show 10 tokens selected for both the progress and feeling sentiment (please note that these aren't in any particular order – instead the table is as a result of words selected across all folds during cross-validation).

Unigram	Bigram	Unigram + Bigram
aaa	accord announced	aaa
surplus	zero percent	abnormal
illegal	capital declined	analysts predict
destroy	company profit	exciting
asian	stock buyback	growing market
litigation	Creditworthiness decreased	Largest technology
embezzlement	stock gained	Lawsuit jpmorgan
examination	wall street	laundering
fine	volatility index	straight year
value	legal claims	breach contract

Figure 5.16 - Words selected for progress classification (manually labelled data)

Unigram	Bigram	Unigram + Bigram
acceptable	aaa credit	abandon
grossing	aaa rated	billion asset
questionable	abc network	cash flow
rallied	income drop	government bond
suppress	percent called	shrank percent
disagree	seek boost	trading stock
distort	later acquire	state law
expense	gain ground	jpmorgan led
investigation	dollar bond	Exclusive
policy	compliance action	development

Figure 5.17 – Words selected for feeling classification (manually labelled data)

We have neglected to include the corresponding tables for the automatically labelled data as automatically labelled data had lower overall accuracy than manually labelled data and was not further used in price prediction. In addition, it should be noted that the words shown in either table are not necessarily exclusive to the table. For example, "jpmorgan led" which appears in the 8 row of the Unigram + Bigram column also appears as a bigram feature when classifying based on progress sentiment.

5.6. Document Classification

5.6.1. Manual Classification

In this section, we detail the results of manual document classification and the settings used to achieve the results. As we use the linear classifier, the parameters that need to be set are the class weights and the cost. Other parameters to be set are default parameters by the classifier.

First, we discuss the classification of progress sentiment and show the results, followed by the classification of feeling sentiment.

5.6.1.1. Progress Sentiment Classification

In order to set the weights, we need to look at the support for each class. Figure 5.18 shows the number of articles supporting each class. The hyper-parameters used for configuring the SVM are as follows:

LinearSVC (the class used for classification) implements the one-versus-rest classifier for multiclass problems. We use a *C* value of 2.9. We use automatically set class weights (which modifies the C-values for each class) for the SVM as Figure 5.18 shows, the classes are not represented equally in the training sets. Using a StratifiedKFold cross validator, we can preserve the percentage of representation for each sample.

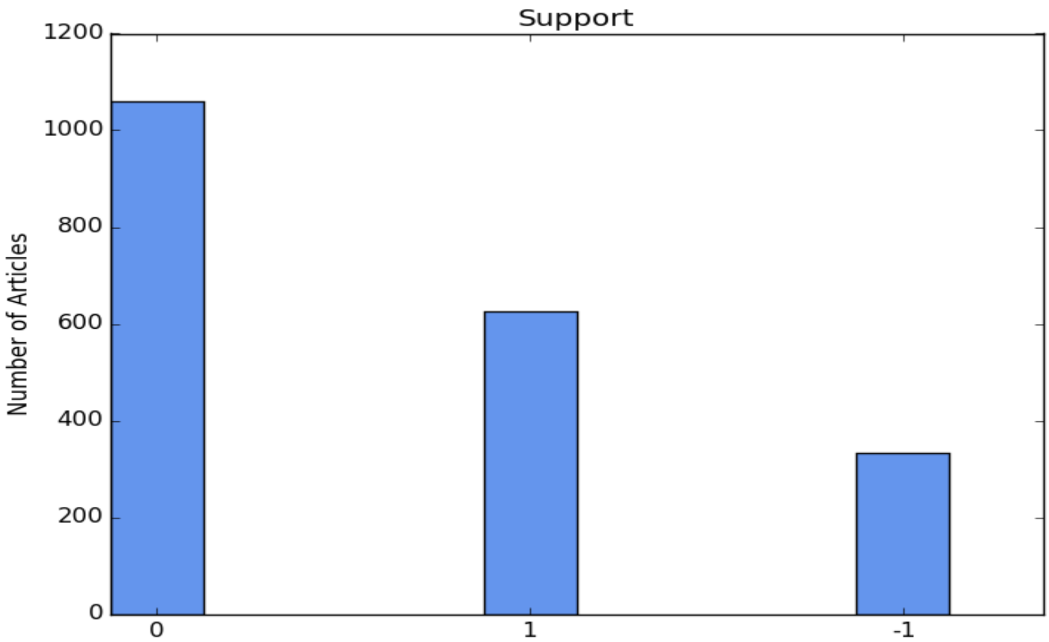

Figure 5.18 – support for the various classes (manual/progress)

Similar settings were used for bigrams, unigrams and combination experiments. In order to determine accuracy, we use cross validation and the following metrics: f-measure, recall, precision and confusion matrix. We compute the average of the scores of all the folds. The confusion matrixes for unigram, bigram and combination (Figure 5.19) show an overview of the accuracy for the three classes.

The confusion matrixes show that there aren't very big differences in the performances of the three methods of tokenisation (except when classifying positive articles). It's very difficult to explain why this is the case except that all three methods carry similar levels of information for this problem.

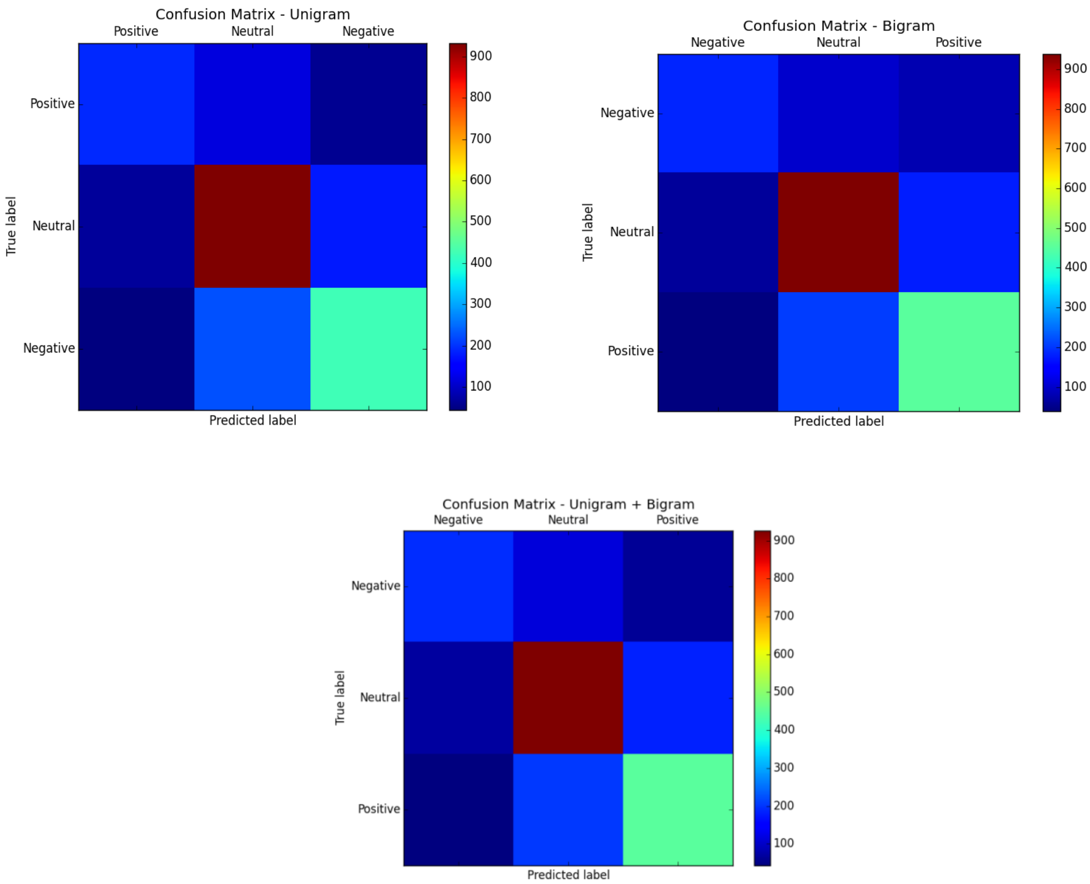

Figure 5.19 – Confusion matrices (Manual/Progress). Top left – Unigram, Top Right – Bigram, Bottom – Unigram + Bigram

	Unigram	Bigram	Unigram + Bigram
F-measure	68.62	69.82	70.51
Recall	68.68	69.97	70.58
Precision	69.04	70.17	70.62

Figure 5.20 – Table of performance of linear SVM measured by cross validation (manual/progress)

Delving into the actual numbers, we see that overall, the bigram does better than the unigram and the combination of both does better than either of them singularly. Combining this information with the confusion matrix, we see that bigrams and the combination perform better due to being able to slightly classify positive news articles better.

Given the similarities in the values, T-tests were performed (with an alpha of 0.05 and a $n - 1$ degree of freedom) to determine there are any significant differences between the results attained with the features. The t-tests confirmed that there are no statistically significant differences between the results.

5.6.1.2. *Feeling Sentiment Classification*

Poorer results were achieved for the classification of feeling sentiment in general. This is contradictory to the initial belief that feeling sentiment would be easier than progress sentiment to classify. We performed classification using a linear SVM as before. The settings for feeling sentiment were quite different. In addition, classification performance for the feeling sentiment was quite poor overall. As per the previous section, we start by introducing the frequencies for the classes (Figure 5.21)

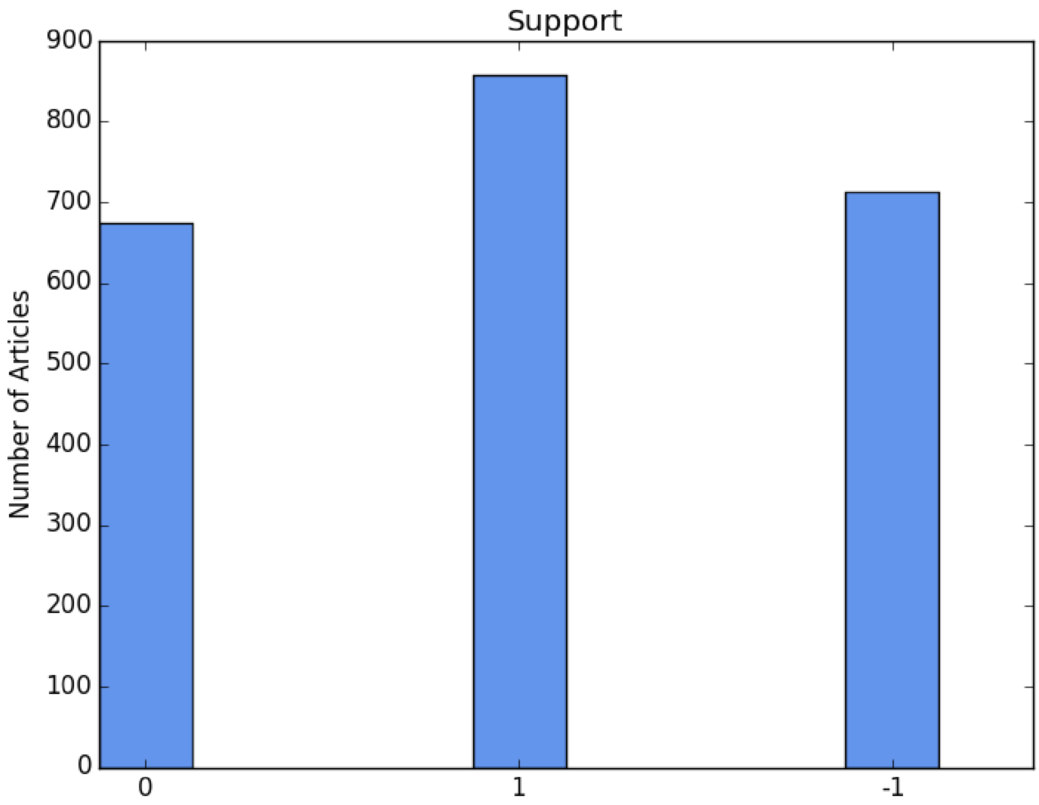

Figure 5.21 – Support for the classes (Manual/ Feeling)

Different hyper-parameter settings were used for classification. The C parameter was set to higher levels with a value of $1 * 10^3$. The other parameters, such as the class weight were also set automatically based on the class.

Considering the confusion matrices (Figure 5.15), we see that the all three methods of tokenising perform very similarly as before. A possible reason for this is that news articles often bear mixed feelings. On the surface, it may seem that news articles bear feeling sentiment orientations that lean towards one way or the other but this isn't so. News articles often carry information that lean to both sides. A classic example of such news articles is articles that discuss "happy" sentiment. In a few of these articles, there's also discussion of past "sad" sentiment that led to perhaps structural changes that result in improvement. Hence, while progress sentiment might be relatively clear, feeling sentiment can often be ambiguous when it comes to classifying neutral articles.

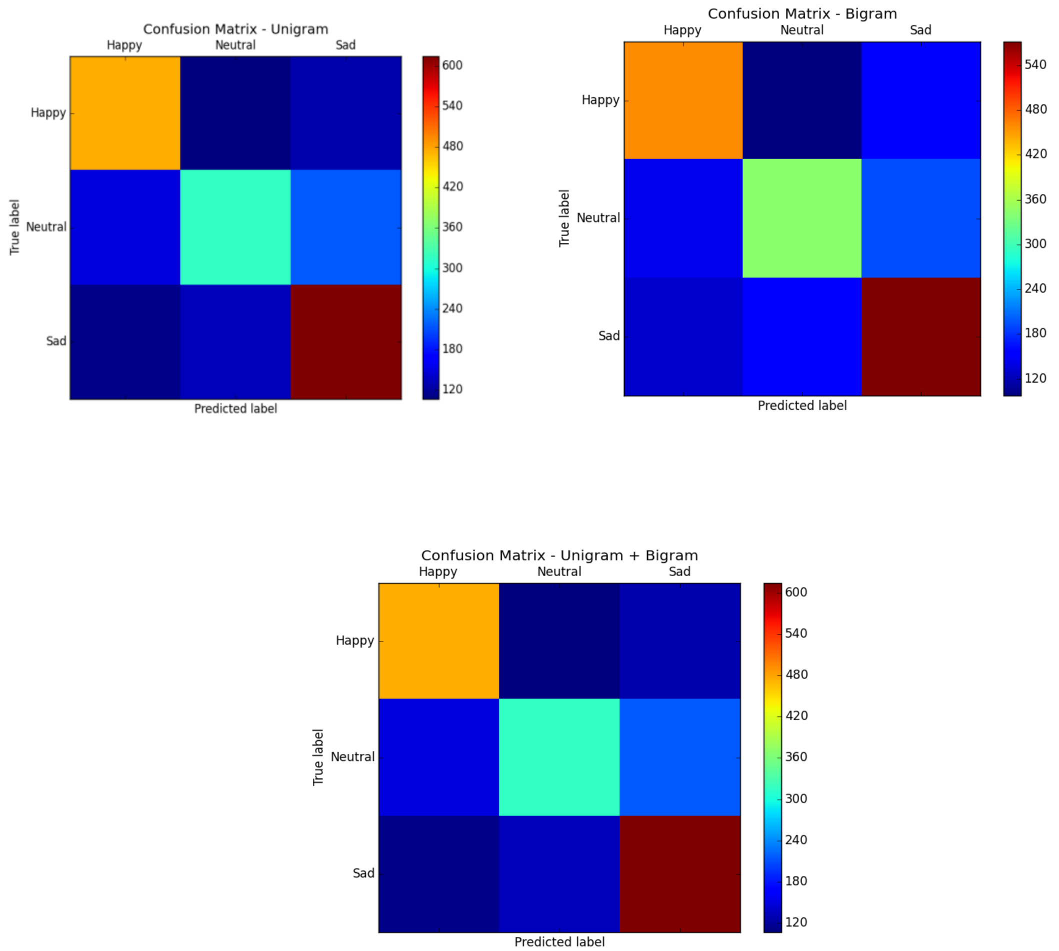

Figure 5.22 – Confusion matrices (Manual/ Feeling). Top left – Unigram, Top Right – Bigram, Bottom – Unigram + Bigram

	Unigram	Bigram	Unigram + Bigram
F-measure	62.00	60.87	63.68
Recall	62.48	61.06	63.92
Precision	62.23	61.15	63.99

Figure 5.23 - Table of performance of linear SVM measured by cross validation (Manual/ Feeling)

Bigrams are typically expected to do better than unigram due to the fact they retain sentence structure but clearly, bigrams doesn't do very well for this problem looking at the performance measures in Figure 5.16. However, combination of both performs better than either but not by much.

5.6.2. Automatic Classification

In this section, we follow the same pattern as in section 5.6.1, with the exception that we only discuss progress sentiment. We use a C value of 3.5. For automatic classification, neutral movements are severely underrepresented (only about 46 news articles were classified neutral); hence, we only consider positive and negative movements.

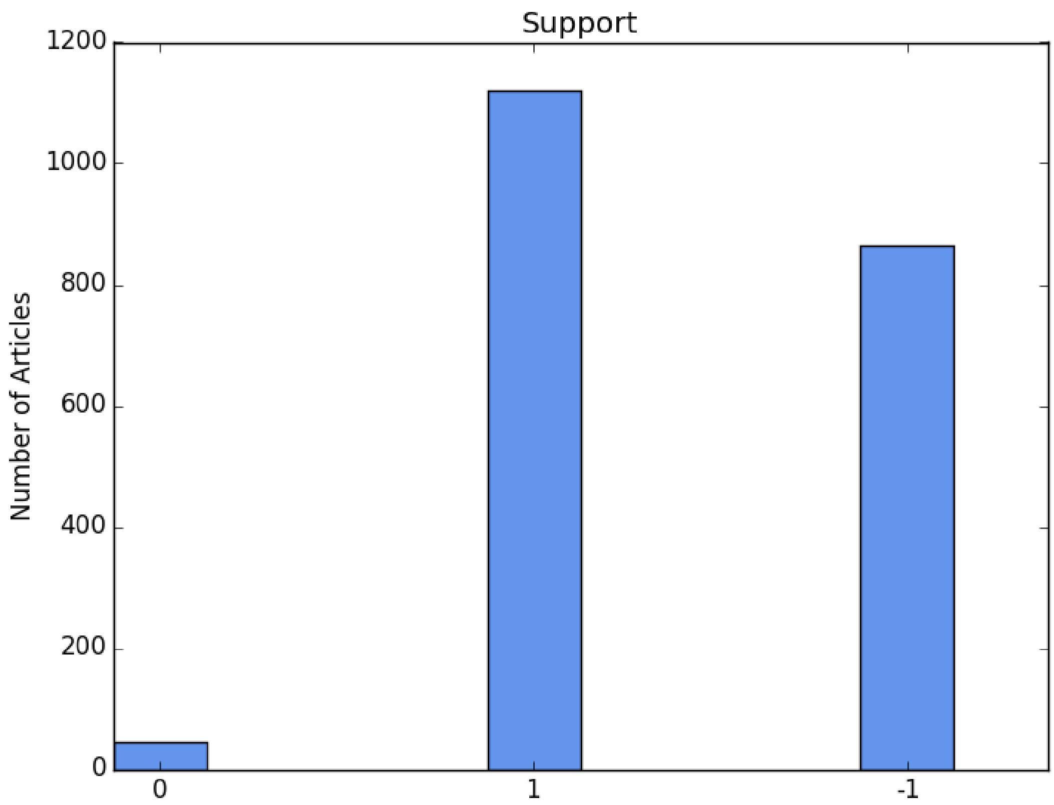

Figure 5.24 – Support for the classes (Automatic / Progress)

Here, we only present the numerical results (Figure 5.25) to restrict repetitiveness as well as the fact that the manually labelled data will be used. It would of course be interesting to consider how well automatically labelled data performs when used subsequently for price prediction, however time restraints prevent this. In addition, the intention was not to use automatic labelling for classification; instead, it provided an adequate benchmark for comparison with the results of manual labelling. We have extensively discussed the pitfalls with automatic labelling (section 5.3.) we believe that the results here can be explained by these.

	Unigram	Bigram	Unigram + Bigram
F-measure	64.50	66.30	64.49
Recall	67.97	72.55	61.36
Precision	61.43	61.12	68.05

Figure 5.25 - Table of performance of linear SVM measured by cross validation (Automatic/ Feeling)

5.7. Price Prediction Results

We present summary statistics of the features in the appendix (excluding statistics) for each of the company analysed section 7.2.

An important aspect of the price prediction is setting the window width. For companies (Chevron, Disney, Microsoft, Visa, J.P. Morgan, Pfizer, Goldman) that show less frequent swings in their trends, we used the entire dataset available to us: That is, we train on 320 days, while predicting for 120 days. For companies that show slightly more rapid swings, we train on 120 days (6 months of data), predicting on 120 (Exxon), and finally, for the most unstable companies over the period we train on 60 days (Cocacola, IBM).

As with sentiment classification, we have opted to use a linear SVM. This again reduces the problem of setting parameters to finding the C value. We employ a grid-search with cross-validation in finding the C-values. The process of finding the C value has to be repeated for every trading day. It's easy to see how the process can be expensive during the experimental phase but daily predictions would be much faster as we'd be searching for one C-value – the value for the current day.

In order to present the results, we normalise the all close prices to a base of 1 (at the start of the 120-day period). The ft (financial times series) line is the return you achieve if you buy at the start of the period and hang on to stocks over the 120-day period. The simple return is calculated using the formula:

$$R_t = \frac{P_t - P_{t-1}}{P_t}$$

Where P_t is the price at time t. The cumulative return over a period of k-days is thus calculated as:

$$1 + P_t(k) = (1 + R_1)(1 + R_2) \dots (1 + R_k)$$

$$= \prod_{t=1}^{k} (1 + R_t)$$

Hence, the return to be gained should one hang on to their stocks will change as the price changes (the trend line is simply a normalised version of the stock price trend line)In order to determine the return of a prediction system, we assume that trades are made based on the prediction. Should the prediction system make a false prediction, trades made based on that prediction result a negative return; however, should it make a correct prediction, trades made based on that prediction result in a positive return. We can thus use the cumulative return formula to generate the return over a period of k-days.

Our experiment consists of two phases: using the SVM-HMM model inclusive of the sentiment data and without – this is in order to make a reasonable evaluation of the effect of sentiment data. Figures 5.26 to 5.45 show the results of both experiments.

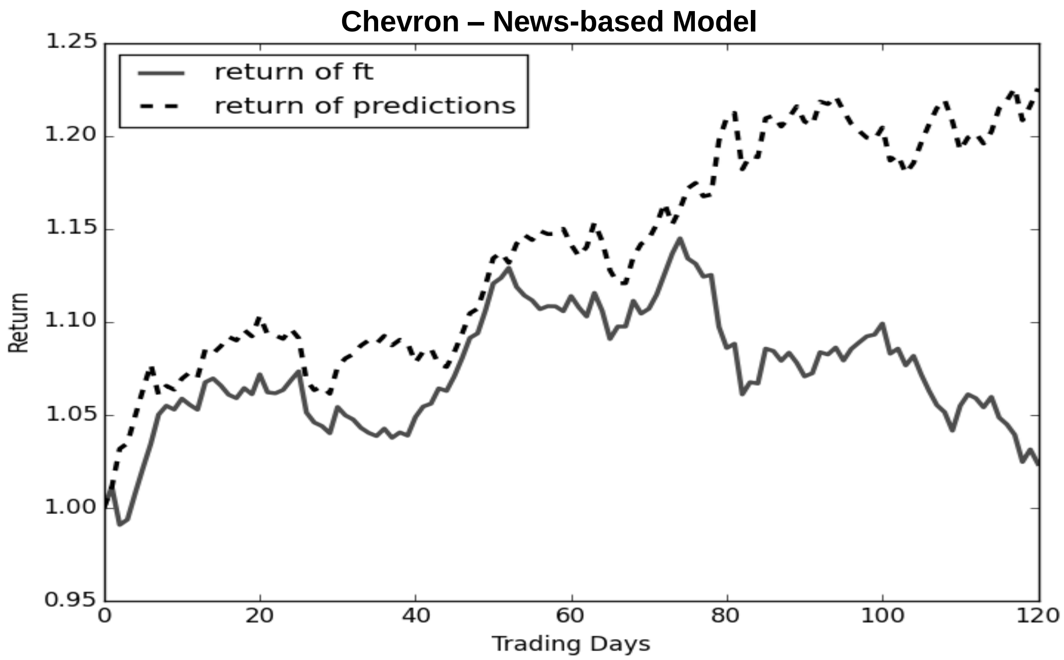

Figure 5.26 –Chevron return based on predictions vs return of time series (inclusive of news-based features)

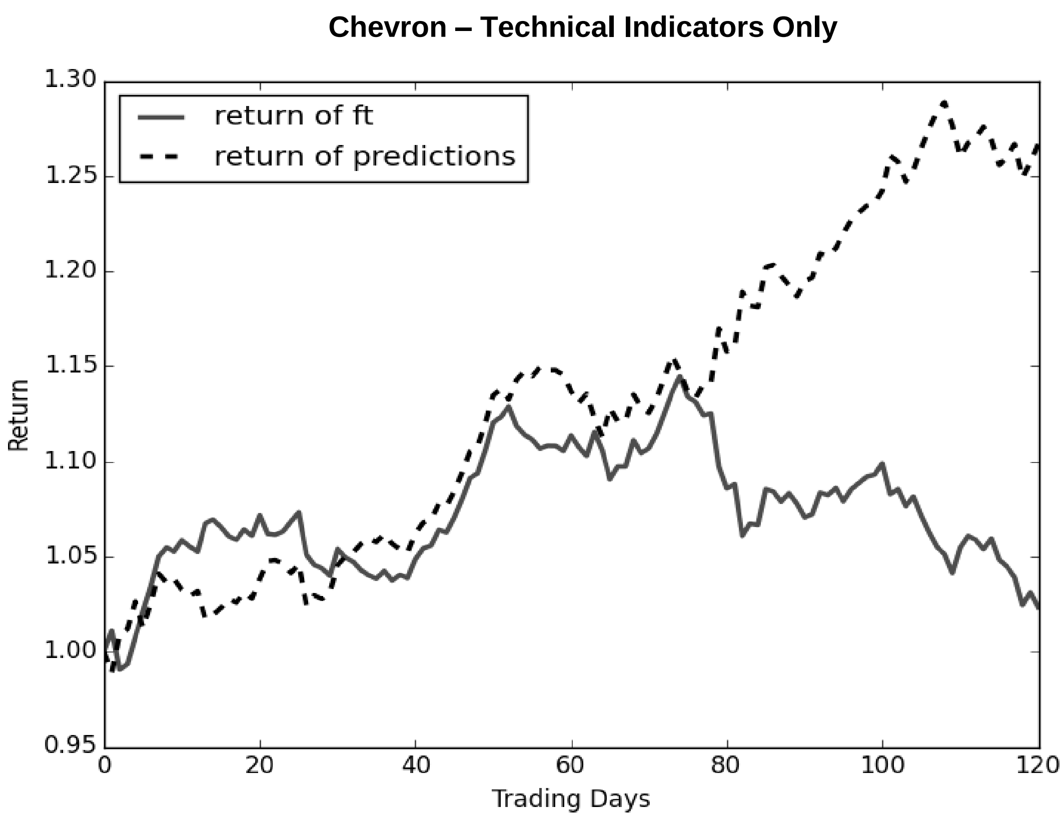

Figure 5.27 - Chevron return based on predictions vs return of time series

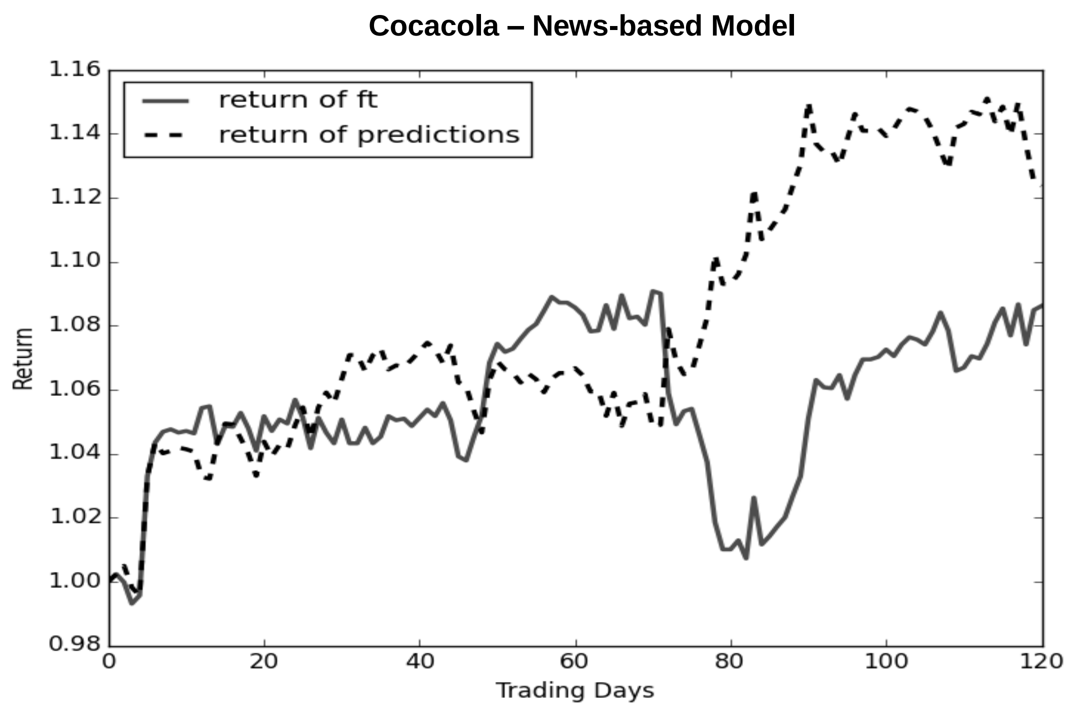

Figure 5.28 - Cocacola return based on predictions vs return of time series (inclusive of news-based features)

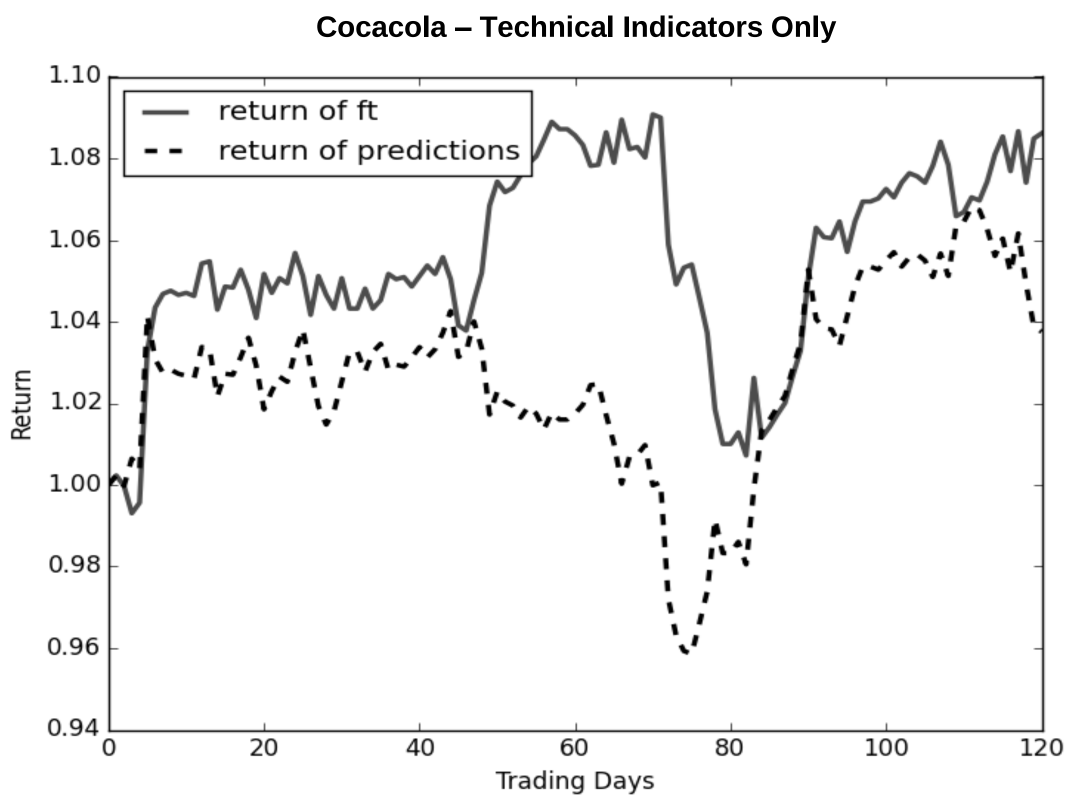

Figure 5.29 - Cocacola return based on predictions vs return of time series

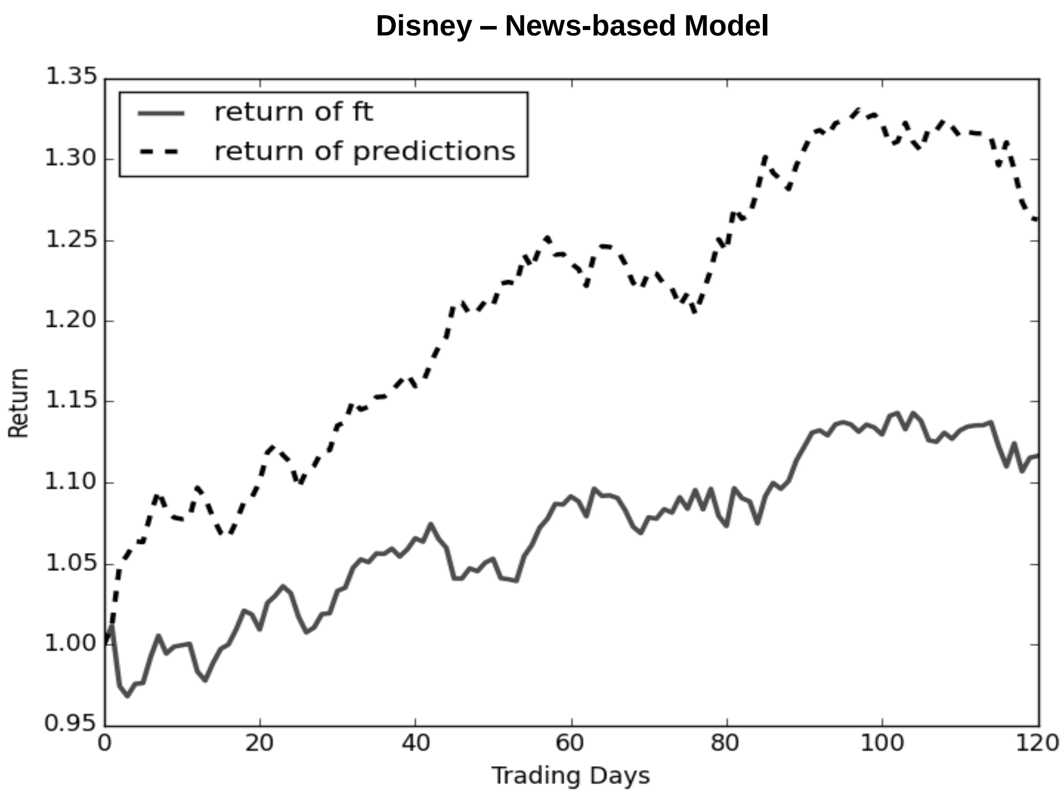

Figure 5.30 - Disney return based on predictions vs return of time series (inclusive of news-based features)

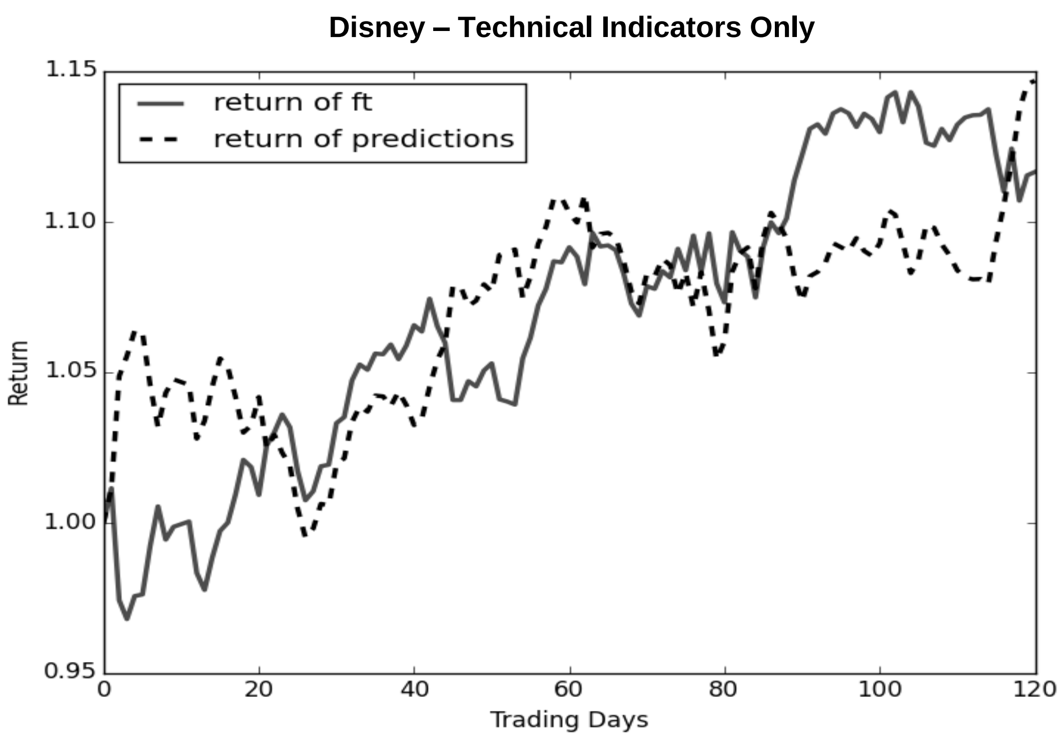

Figure 5.31 - Disney return based on predictions vs return of time series

Exxon – News-based Model

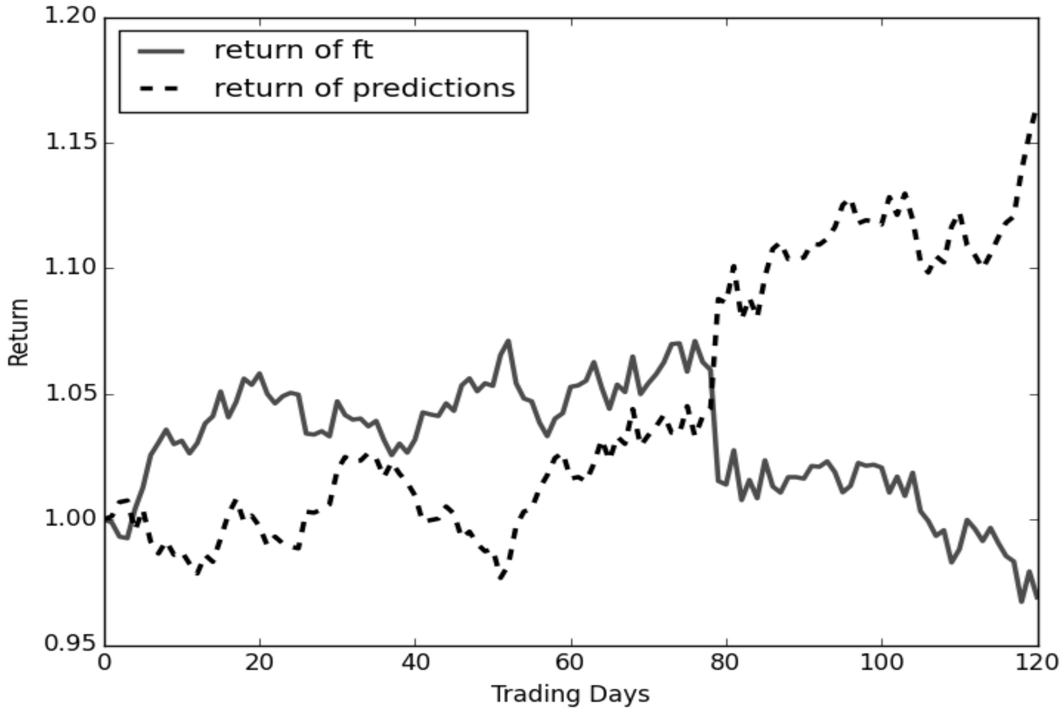

Figure 5.32 - Exxon return based on predictions vs return of time series (inclusive of news-based features)

Exxon – Technical Indicators Only

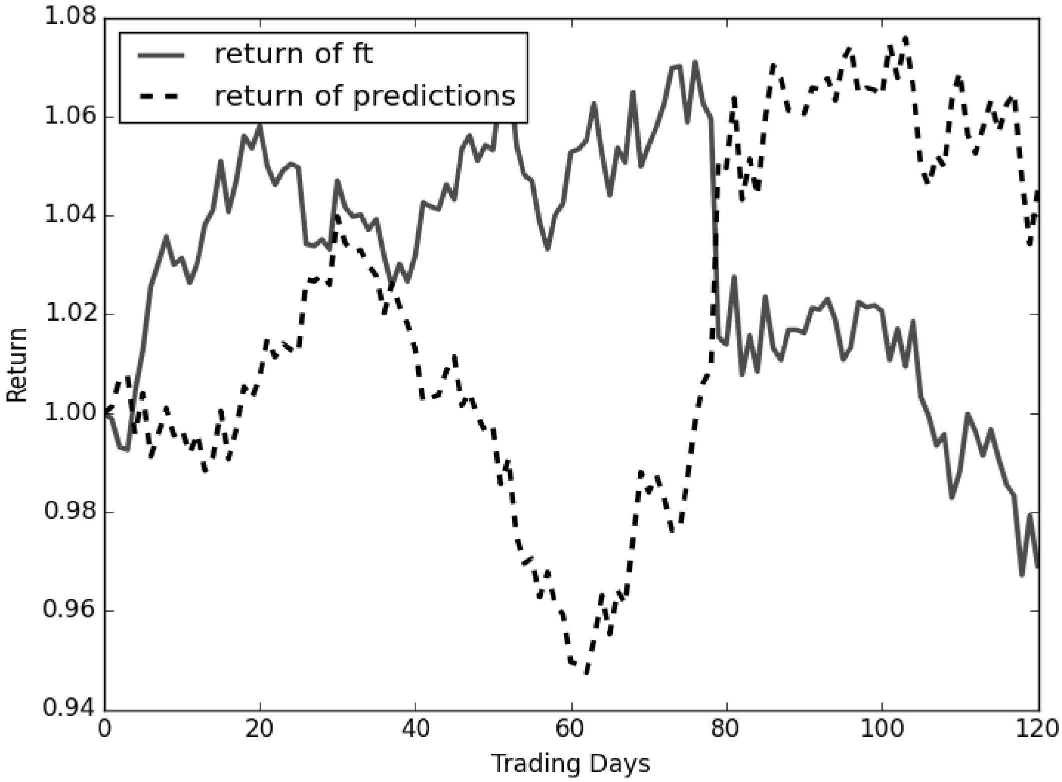

Figure 5.33 - Exxon return based on predictions vs return of time series

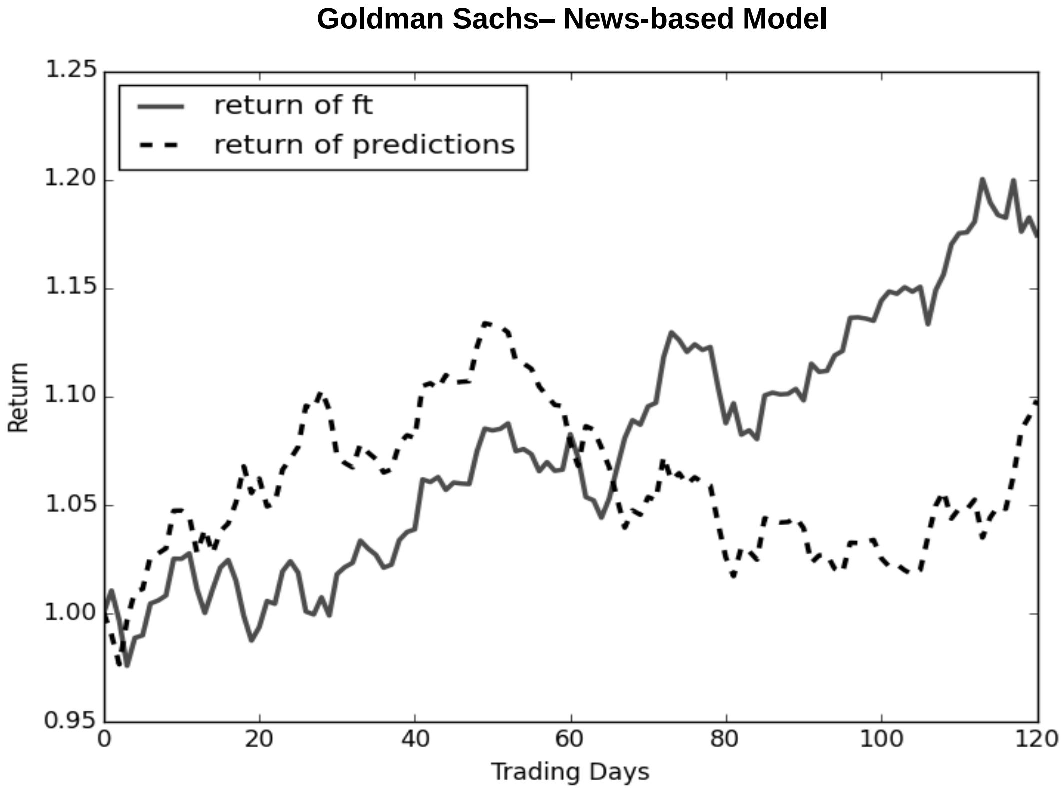

Figure 5.34 – Goldman Sachs return based on predictions vs return of time series (inclusive of news-based features)

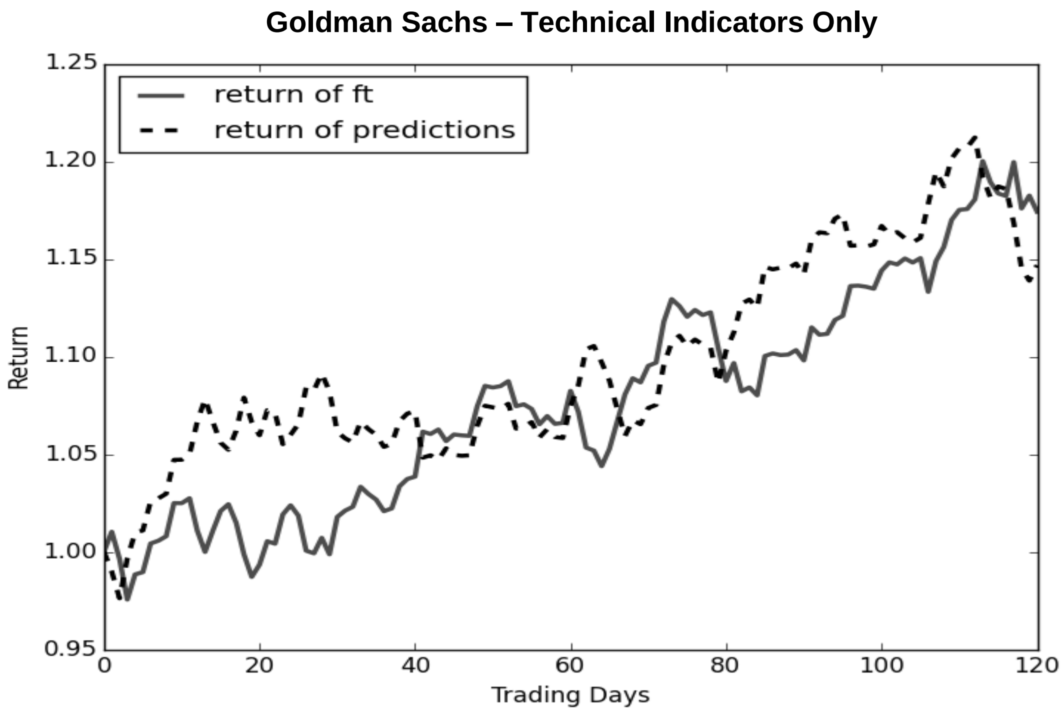

Figure 5.35 – Goldman Sachs return based on predictions vs return of time series

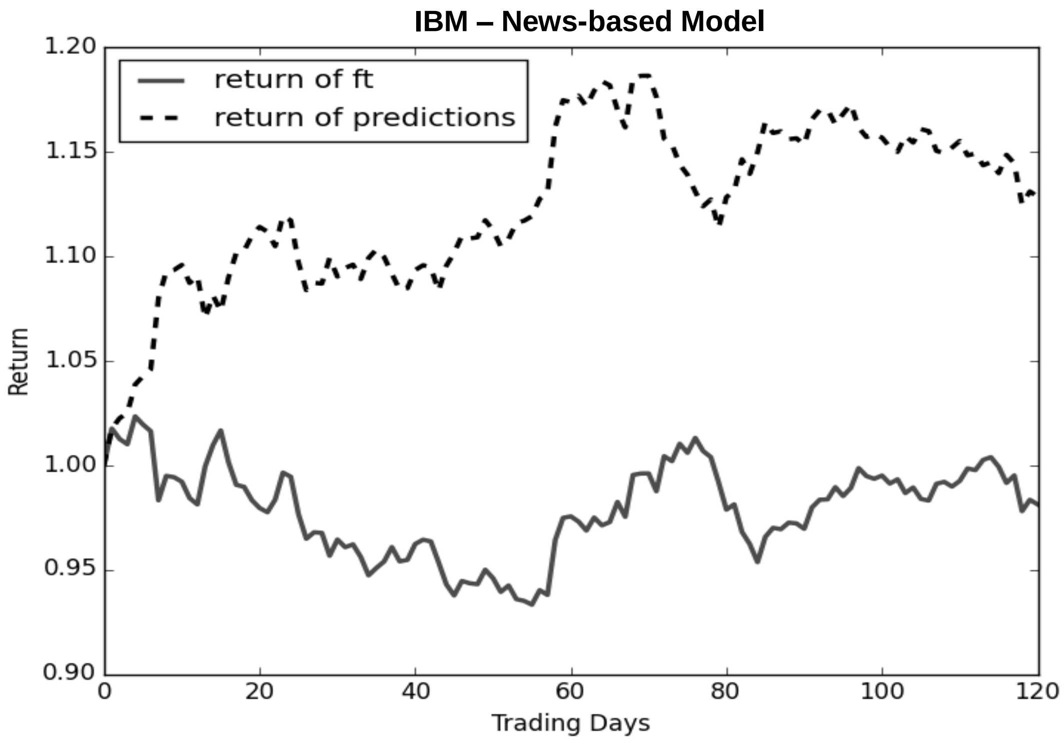

Figure 5.36 - IBM return based on predictions vs return of time series (inclusive of news-based features)

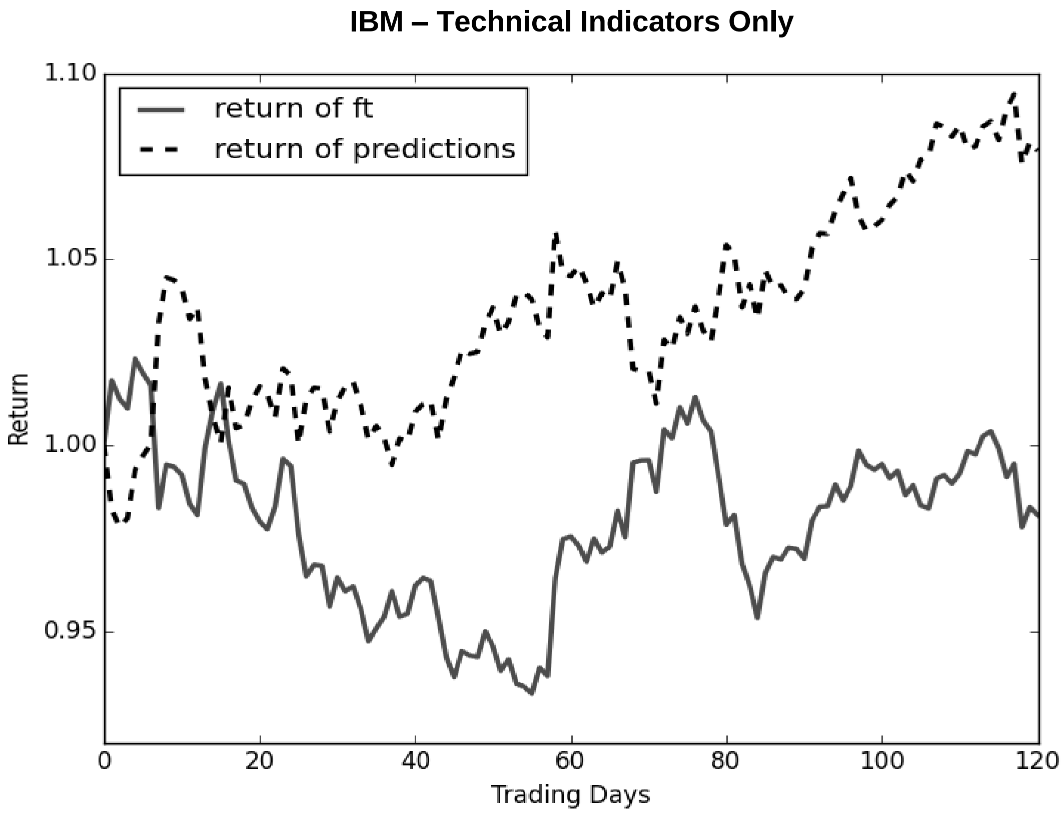

Figure 5.37 - IBM return based on predictions vs return of time series

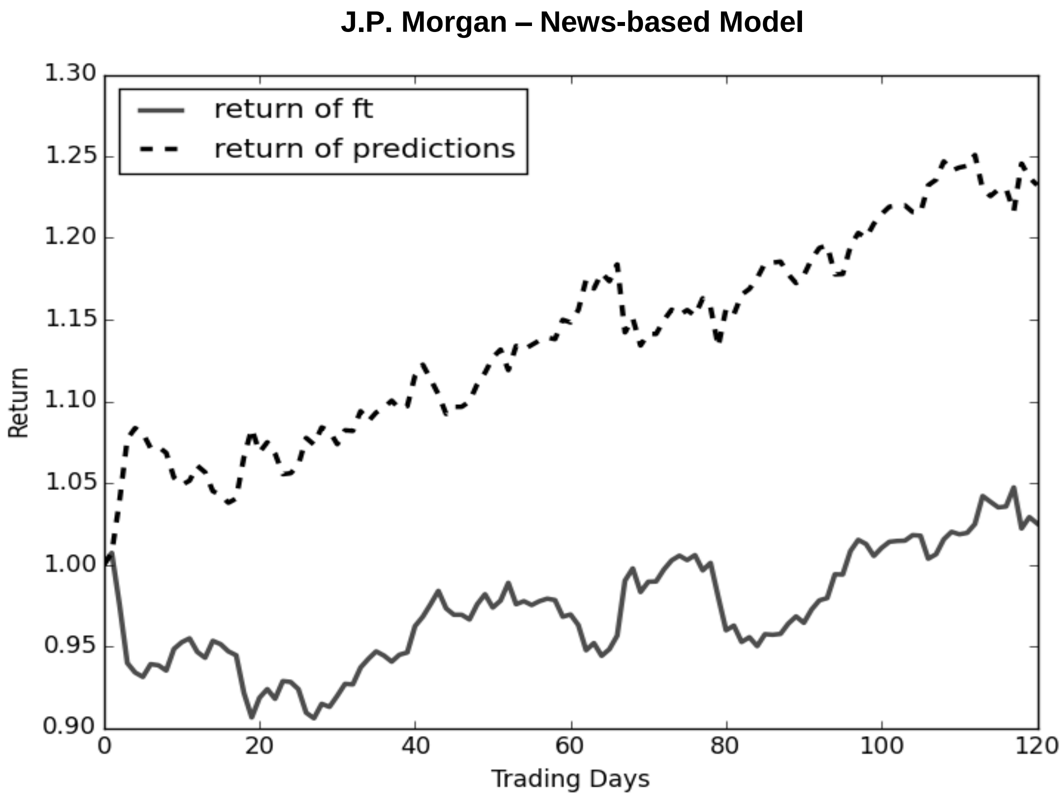

Figure 5.38 – J.P. Morgan return based on predictions vs return of time series (inclusive of news-based features)

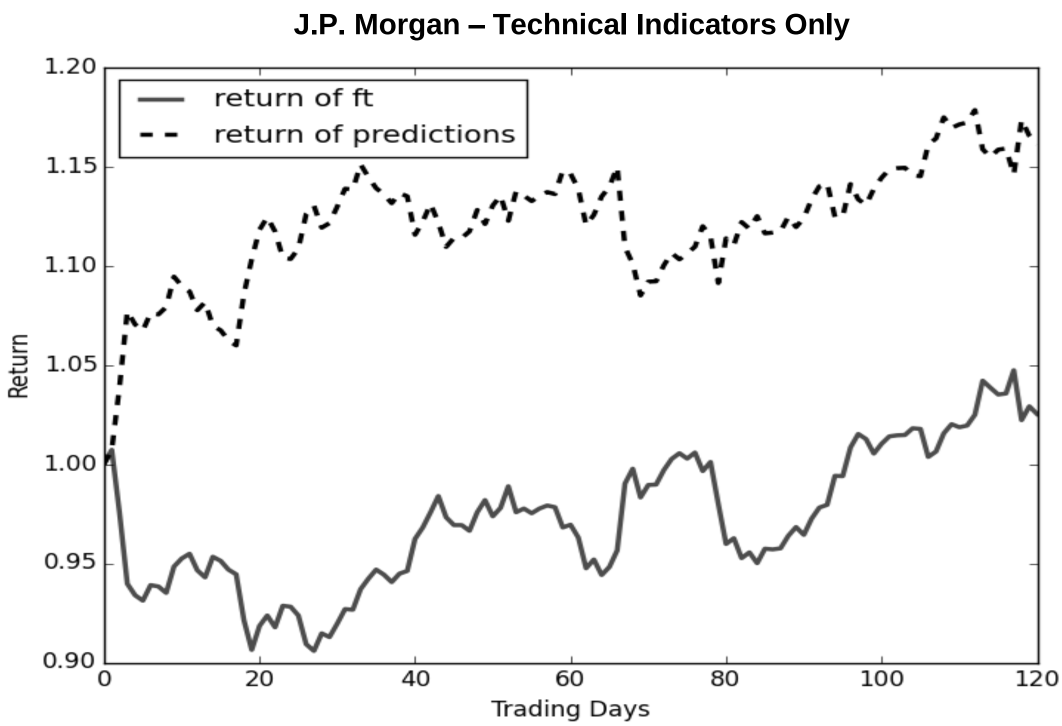

Figure 5.39 – J.P. Morgan return based on predictions vs return of time series

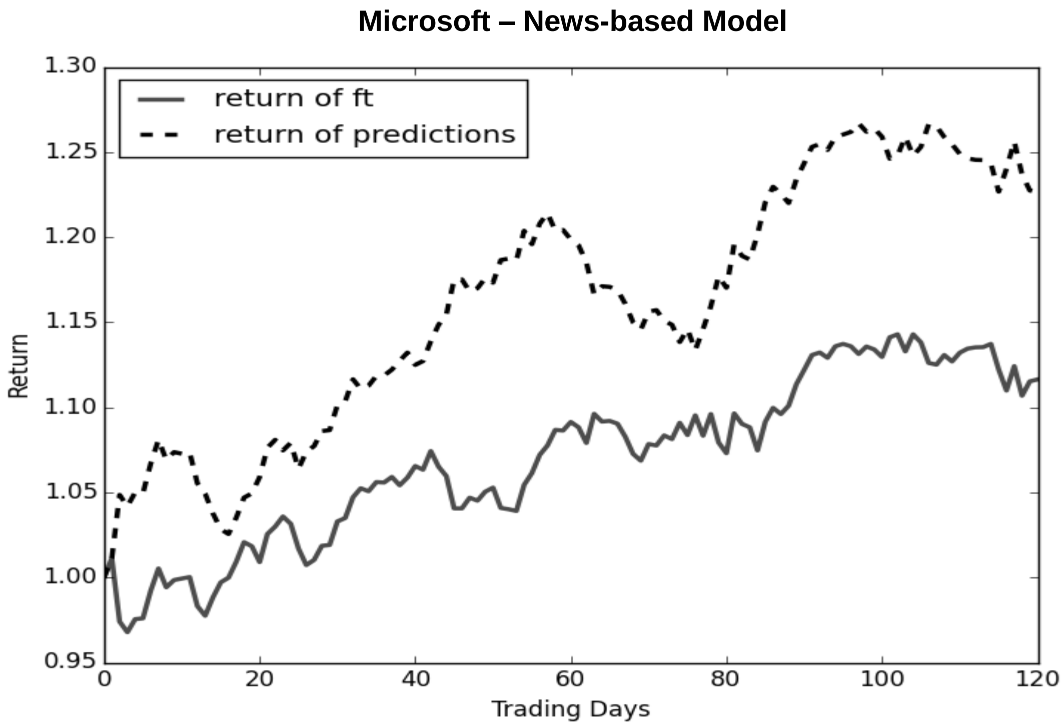

Figure 5.40 - Microsoft return based on predictions vs return of time series (inclusive of news-based features)

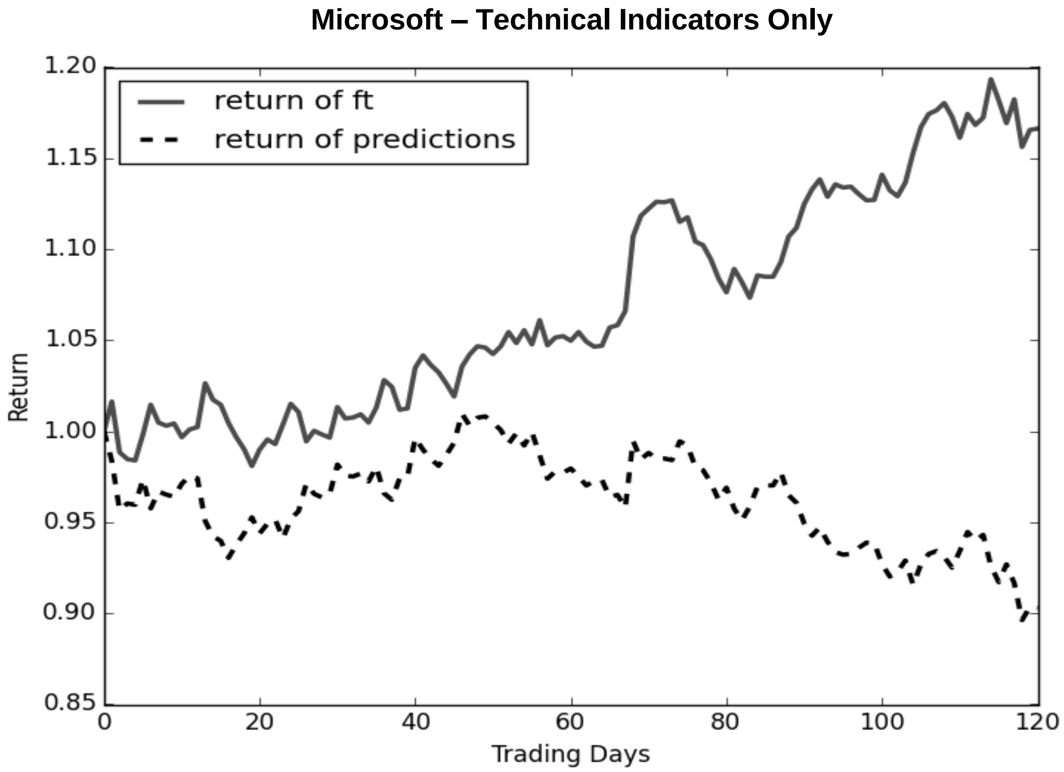

Figure 5.41 - Microsoft return based on predictions vs return of time series

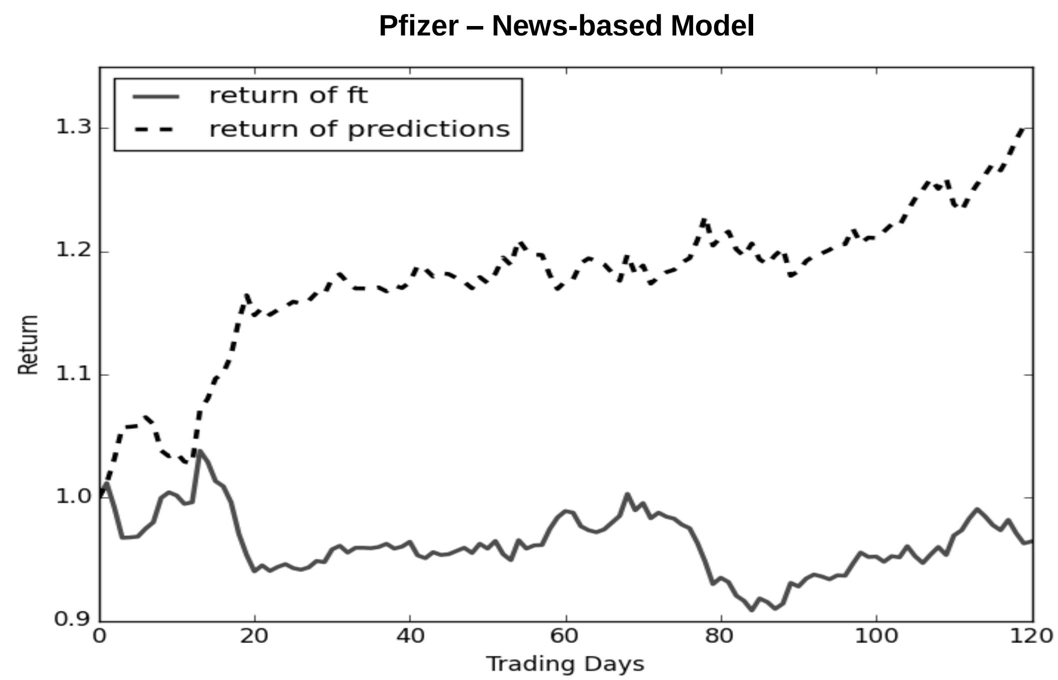

Figure 5.42 - Pfizer return based on predictions vs return of time series (inclusive of news-based features)

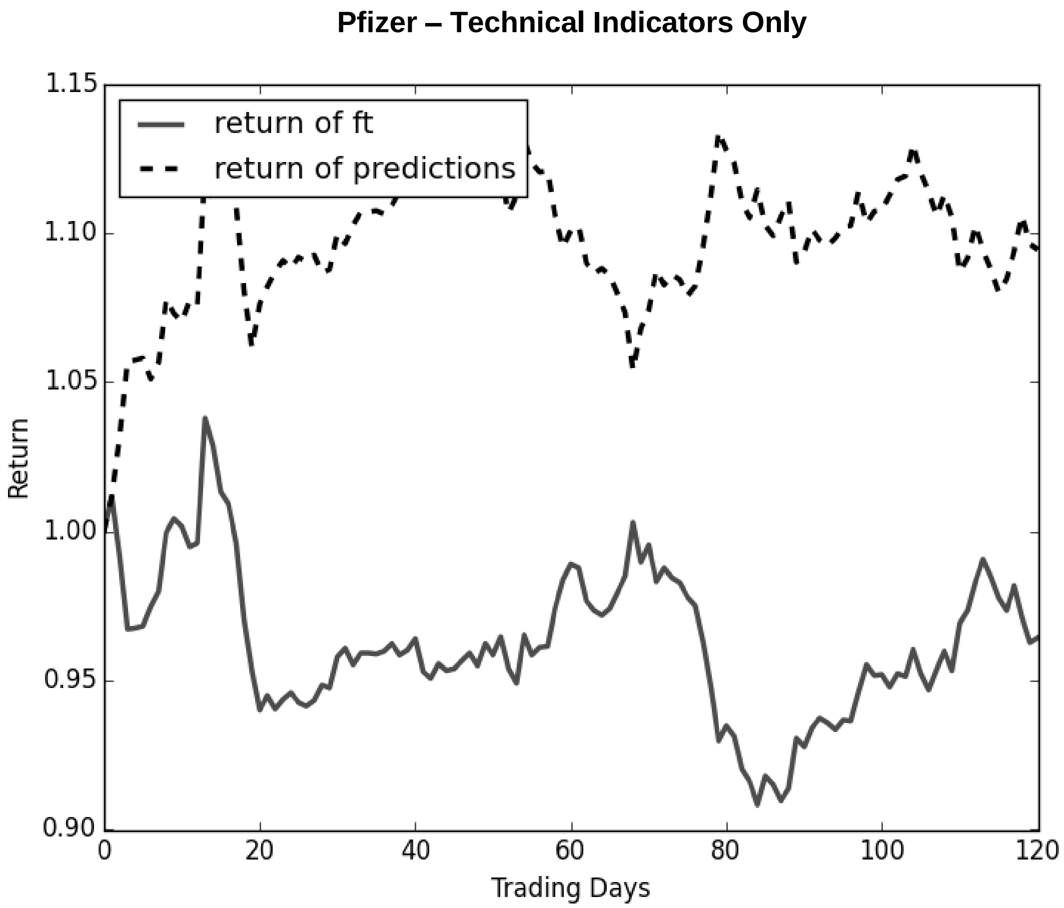

Figure 5.43 - Pfizer return based on predictions vs return of time series

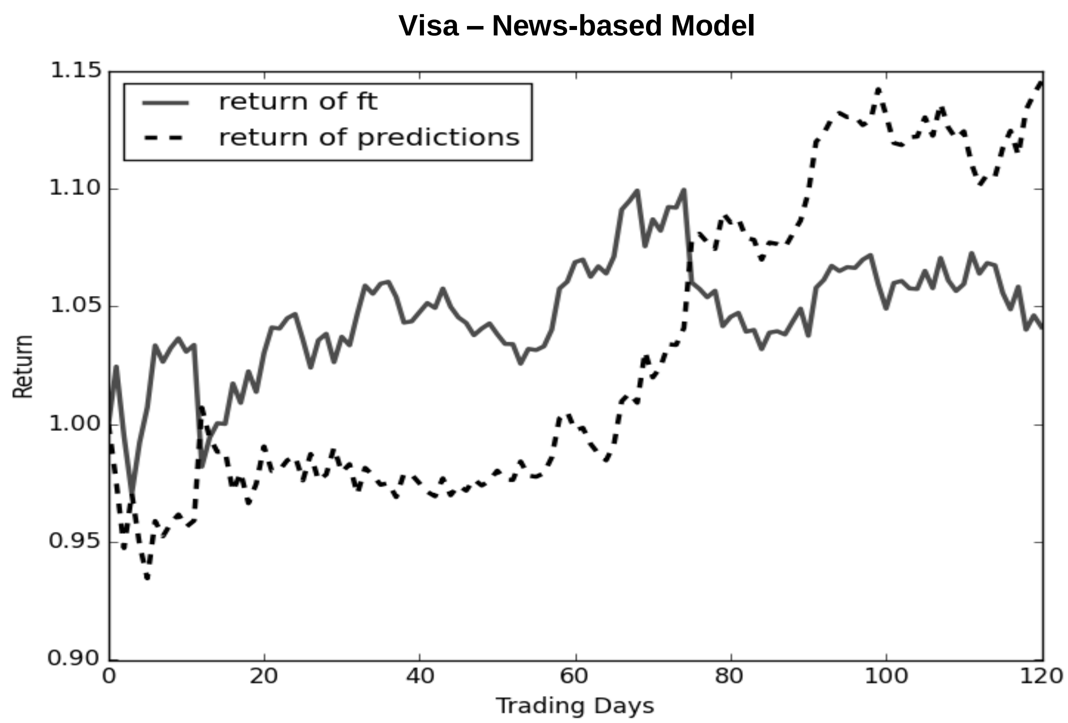

Figure 5.44 - Visa return based on predictions vs return of time series (inclusive of news-based features)

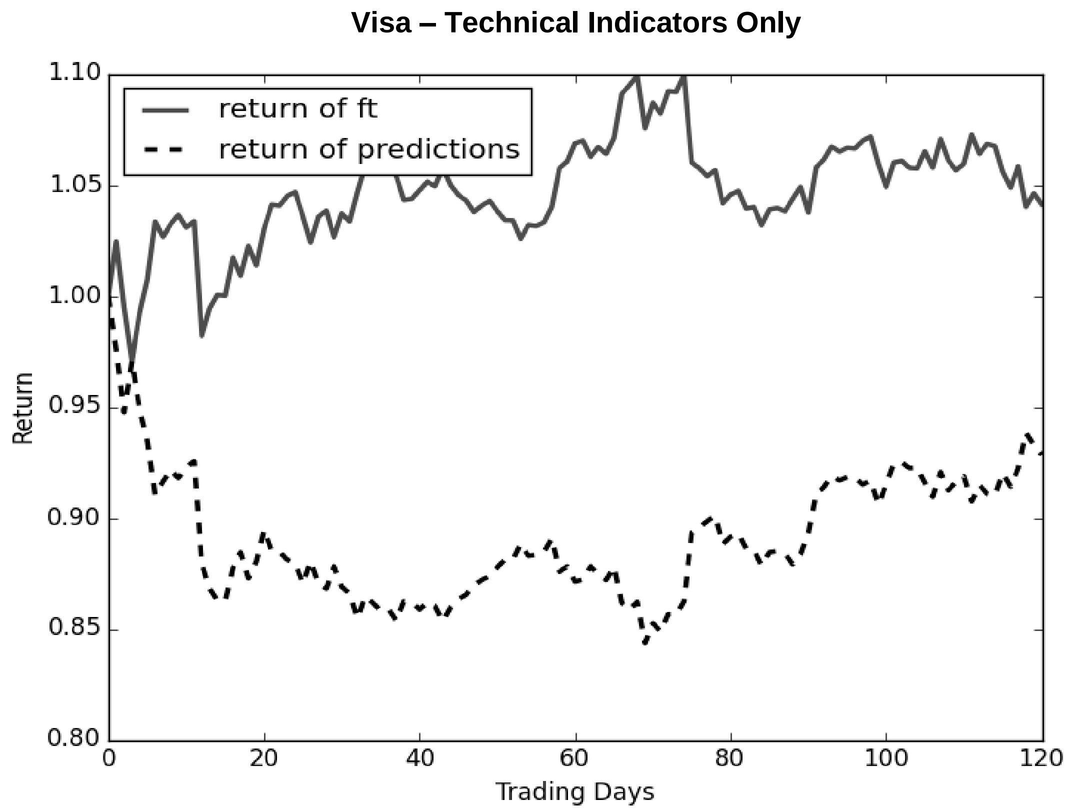

Figure 5.45 - Visa return based on predictions vs return of time series

Examining, all the preceding images, it's quite clear that the addition of news-based features has a real and tangible effect on the overall prediction accuracy of the model. We attempt at this juncture to provide reasons and possible justification for our results. It's important to note that each trend line is different and we have treated each company differently, and thus each of the models have different results.

Chevron and Goldman's news based models, experienced poorer performance than the technical based models. We propose that the reason for this is the less than perfect classification of sentiment as the other news based models experienced a better performance both than the market and the technical-only models.

We highlight Cocacola seems to be a special case of (relative) positivity given that the very little news, that the classifier has to work with. We also note that it performs better than the technical indicators only model.

In addition, we note that though the graphs seem to say otherwise, the technical-only indicators had higher than random hit ratios, ranging from 51.67% (Visa) – 61.67 Chevron), which correlates with the results in the literature. Our graphical way of showing the results, however is a better way of showing the effects of the models over time.

In general, we see that both technical indicator only models and news-based model tend to suffer a loss in performance at the same time, however, the news based model tends to suffer less and recover faster. We believe that the results aren't only dependent on news, but also on explicitly stating the sentiment for days that there isn't any news (as discussed in section 5.3.1.)

In conclusion, taking the hit ratios, into account, we note that adding news seems to have a stabilising effect on predictions so that highly fluctuating prices have less adverse effect on prices. Given, our 80% success rate (in terms of predictability of news vs technical indicators), we can say that our results correlate with the results of the literature review: in general news can be a reasonable basis for stock price prediction.

6. Conclusion

We hope it's say that the project is overall a success as both phases (sentiment analysis and price prediction) were completed successfully. One important result to point out is that one needn't necessarily achieve sentiment classification accuracy of nearly 100% in order to base predictions based on automatic classification. We however, spend some time discussing what could be improved in order to make the system more reliable

6.1. Future Work

One of the first obstacles to work around, as is the case in many projects is time restrictions. This severely restricted the amount of time we could spend manually classifying and the results of Goldman (section 5.3.3.) show us that automatic labelling while they can perform well, have to be used on a perfectly curated data set – which again requires a lot of manual work. The manual data was also classified primarily by an amateur despite the relative lack of knowledge compared to an expert, it has shown good results. Regardless it is recommended that further work utilise financial experts to manually classify the dataset. It's further recommended that any manually classified data set is further reviewed by a separate team of experts. Given only this recommendation, it's easy to see how this recommendation could make the process of developing a more advanced system could be significantly more expensive than the cost of this project in monetary terms (which was the cost of a 6-month Financial Times subscription).

Any issue that was the repercussion of having little time is the dataset. Our meagre ~2250-article dataset proved to be useful enough but an even better dataset would contain articles upwards of 30000. This can easily be achieved by pooling news articles from several news sources. Another benefit of having such a large dataset is customisation of sentiment classification on a per-organisation or at least per-industry scale. We hope that the expected improvement to the sentiment classification accuracy is by this point obvious to the user. As a test, we decided to cross-validate the Goldman Sachs news dataset ((unigrams + bigrams) feature set and progress-sentiment classification) and we achieved an f-measure of 75.09%, significantly better than the 70.51% of the pooled data set.

We mentioned in section 4.3. that the means by which the window width is decided is by evaluating the linear approximation trends manually, which of which the result was a rigid window width; however, upon reflection, a fixed training window is too rigid, it's better to integrate piecewise linear approximation into the overall process in order that training occurs with a customised sliding window. Hence, for the prediction based for test data x_t at current time t the training window w of length m, would only include training data in a segment J defined by piecewise linear approximation. With the constraint that x_t is the last known point in the time series defined by J,

6.2. External Aspect

A research paper accompanying this dissertation was written titled "Sentiment Analysis for Stock Price Prediction". The paper was co-authored with Michel Valstar. The paper was written successfully for the fulfilment of the external aspect of the current work.

6.3. Personal Reflection

The current dissertation has been less bumpy of a ride than I thought it'd be. It's been a great learning experience on my end because I was given as much elasticity as possible intellectually by my supervisor. I started out with knowing nothing about the current field – it's safe to say that the only aspects of the current work that I had any prior knowledge of were Support Vector Machines and Hidden Markov Models – everything else has been learnt as I went along.

The hardest part of the project was initially attempting to recruit financial experts. An important lesson learnt is that financial experts are very unlikely to work for free – especially not work that hopes to put them out of work in the future. Regardless of every obstacle and delay, I believe that the project was a success in every sense of the word.

7. Appendix

7.1. Keys for Transforming Sentiments to Numbers

7.1.1. Progress Sentiment

Sentiment	Numerical Value
Up	-1
Neutral	0
Down	1

7.1.2. Feeling Sentiment

Sentiment	Numerical Value
Sad	-1
Neutral	0
Happy	1

7.2. Summary Statistics for Each Company

7.2.1. Chevron

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %K	0.00	100.00	56.92	31.62
Stochastic %D	2.59	98.95	56.94	29.57
Slow Stochastic %D	3.94	98.40	56.98	28.74
Momentum	-9.31	10.63	0.43	4.04
Rate of Change	92.17	108.69	100.42	3.34
William's %R	0	-100	-43.07	31.62
A/D Oscillator	-1.12	1.82	0.52	0.50
Disparity5	96.67	102.80	100.05	0.95
Disparity10	95.24	103.82	100.12	1.53
Price Oscillator	-0.02	0.02	0.0008	0.008
Commodity Channel	-236.24	276.01	19.70	112.89
Relative Strength	82.83	82.83	52.86	11.71

Figure 7.1 – Descriptive Statistics for Technical Indicators (Chevron)

7.2.2. Cocacola

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic $\%K$	0.00	100.00	57.82	30.34
Stochastic $\%D$	0.51	99.88	57.66	27.88
Slow Stochastic $\%D$	1.51	99.15	57.51	27.06
Momentum	-3.09	2.66	0.14	1.14
Rate of Change	92.71	106.99	100.41	2.28
William's %R	0.00	-100	-42.17	30.34
A/D Oscillator	-1.49	1.98	0.50	0.48
Disparity5	96..62	103.01	100.07	0.94
Disparity10	95.48	104.12	100.15	1.38
Price Oscillator	-0.02	0.02	0.0006	0.007
Commodity Channel Index	-358.84	347.35	20.60	112.41
Relative Strength Index	23.00	75.98	52.66	11.14

Figure 7.2 - – Descriptive Statistics for Technical Indicators (Cocacola)

7.2.3. Disney

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic $\%K$	0.00	100	62.48	29.93
Stochastic $\%D$	2.05	99.32	62.49	27.31
Slow Stochastic $\%D$	3.66	98.3	62.53	26.30
Momentum	-5.37	10.14	1.17	2.51
Rate of Change	91.93	114.49	101.82	3.74
William's %R	-100.00	0.00	-37.51	29.93
A/D Oscillator	-0.89	2.55	0.58	0.449
Disparity5	104.66	104.66	100.26	1.19
Disparity10	95.94	106.52	100.60	1.80
Price Oscillator	-0.02	0.03	0.003	0.01
Commodity Channel Index	-248.09	298	48.03	102.73
Relative Strength Index	30.85	82.93	58.08	10.01

Figure 7.3 – Descriptive Statistics for Technical Indicators (Disney)

7.2.4. Exxon

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic $\%K$	0.17	100	54.16	31.11
Stochastic $\%D$	1.02	97.37	54.16	29.04
Slow Stochastic $\%D$	1.83	95.97	54.15	28.31
Momentum	-9.36	6.94	0.24	2.98
Rate of Change	90.54	107.54	100.31	3.17
William's %R	-99.83	0	-45.83	31.11
A/D Oscillator	-0.69	1.64	0.49	0.48
Disparity5	96.38	102.97	100.04	0.91
Disparity10	94.78	103.76	100.10	1.43
Price Oscillator	-003	0.02	0.0004	0.008
Commodity Channel	-292.76	269.53	12.08	109.58
Relative Strength	18.88	77.38	51.88	11.48

Figure 7.4 – Descriptive Statistics for Technical Indicators (Exxon)

7.2.5. Goldman

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic $\%K$	0	100	58.95	31.28
Stochastic $\%D$	2.66	97.83	59.00	29.43
Slow Stochastic $\%D$	3.91	96.93	59.02	28.66
Momentum	-16.99	17.06	1.76	7.19
Rate of Change	90.50	113.24	101.26	4.63
William's %R	-100	0	-41.04	31.28
A/D Oscillator	-0.95	2.32	0.54	0.50
Disparity5	95.90	104.10	100.17	1.35
Disparity10	93.99	105.86	100.39	2.06
Price Oscillator	-0.03	0.03	0.0019	0.01
Commodity Channel	-279.82	281.85	30.23	108.09
Relative Strength	29.11	80.11	55.30	11.83

Figure 7.5 – Descriptive Statistics for Technical Indicators (Goldman)

7.2.6. IBM

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %K	0	100	50.57	31.11
Stochastic %D	0.8	98.74	50.58	29.05
Slow Stochastic %D	2.88	97.78	50.60	28.19
Momentum	-24.83	18.29	-0.03	7.00
Rate of Change	88.32	109.26	100.05	3.63
William's %R	-100	0	-49.42	31.11
A/D Oscillator	-2.2	2.39	0.50	0.56
Disparity5	92.41	104.29	99.99	1.19
Disparity10	91.12	105.19	100.00	1.78
Price Oscillator	-0.05	0.03	7e-20	0.01
Commodity Channel	-497.3	430.24	1.64	114.40
Relative Strength	22.88	76.67	50.30	10.76

Figure 7.6 – Descriptive Statistics for Technical Indicators (IBM)

7.2.7. JP Morgan

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %K	0	100	60.30	30.73
Stochastic %D	4.76	98.15	60.33	28.54
Slow Stochastic %D	6.39	96.9	60.35	27.62
Momentum	-5.64	6.66	0.50	2.22
Rate of Change	90.7	113.6	101.06	4.13
William's %R	-100	0	-39.69	30.73
A/D Oscillator	-0.9	2.58	0.54	0.50
Disparity5	95.4	103.34	100.12	1.26
Disparity10	93.39	104.91	100.34	1.94
Price Oscillator	-0.04	0.03	0.001	0.01
Commodity Channel	-328.71	246.43	29.14	106.92
Relative Strength	26.61	80.07	54.81	11.51

Figure 7.7 – Descriptive Statistics for Technical Indicators (JPMorgan)

7.2.8. Microsoft

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %*K*	2.22	100	63.27	26.16
Stochastic %*D*	6.51	98.78	63.23	23.56
Slow Stochastic %*D*	8.12	97.56	63.22	22.49
Momentum	-4.5	4.92	0.57	1.37
Rate of Change	87.56	117.07	101.70	3.99
William's %R	-97.78	0	-36.72	26.16
A/D Oscillator	-2.3	3.13	0.50	0.54
Disparity5	89.7	107.41	100.25	1.51
Disparity10	89.78	107.65	100.57	2.11
Price Oscillator	-0.07	0.04	0.003	0.012
Commodity Channel	-265.14	450.17	52.86	99.14
Relative Strength	29.47	79.85	56.80	9.67

Figure 7.8 – Descriptive Statistics for Technical Indicators (Microsoft)

7.2.9. Pfizer

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %*K*	0	100	56.64	30.10
Stochastic %*D*	4.01	97	56.62	27.97
Slow Stochastic %*D*	4.96	95.86	56.57	27.22
Momentum	-2.98	2.51	0.14	1.04
Rate of Change	90.7	108.91	100.57	30.10
William's %R	-100	0	-43.35	3.50
A/D Oscillator	-1.14	1.9	0.53	0.46
Disparity5	96.01	103.56	100.08	1.11
Disparity10	94.57	104.59	100.18	1.72
Price Oscillator	-0.04	0.02	0.0006	0.009
Commodity Channel	-307.87	310.55	27.52	110.72
Relative Strength	25.32	80.83	52.88	10.90

Figure 7.9 – Descriptive Statistics for Technical Indicators (Pfizer)

7.2.10. Visa

Technical Indicator	Min	Max	Mean	Standard Deviation
Stochastic %K	0	100	57.95	29.14
Stochastic %D	4.62	97.92	58.06	26.30
Slow Stochastic %D	5.59	96.41	58.15	25.28
Momentum	-21.81	21.81	1.91	7.57
Rate of Change	112.32	112.32	101.14	3.97
William's %R	-100	0	-42.04	29.14
A/D Oscillator	2.17	2.17	0.55	0.50
Disparity5	104.71	104.71	100.15	1.30
Disparity10	106.26	106.26	100.36	1.91
Price Oscillator	-0.03	0.03	0.0017	0.01
Commodity Channel	-300.54	334.46	30.89	108.34
Relative Strength	26.45	77.62	54.74	9.84

Figure 7.10 – Descriptive Statistics for Technical Indicators (Visa)

8. Bibliography

Aggarwal, C. C., & ChengXiang, Z. (2012). *Mining Text Data.* Springer Science & Business Media.

Bar-Heim, R., Dimur, E., Feldman, R., Fresko, M., & Goldstein, G. (2011). Identifying and following expert investors in stock microblogs. *Proceedings of the Conferenec on Empirical methods in Natural Language Processing* (pp. 1310-1319). 2011.

Bollen, J., Mao, H., & Zeng, X. (2011). Twitter Mood Predicts the Stock Maret. *Journal of Computational Science*, 1-8.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent sematic analysis. *JASIS*, 391-407.

Deng, S., Mitsubuchi, T., Shioda, K., Shimada, T., & Sakurai, A. (2011). Combining Technical Analysis with Sentiment Analysis for Stock Price Prediction. *Dependable, Autonomic and secure Computing (DASC)*, 800-807.

Donald, L. (1980). Commodities.

Elkan, C. (1999). Notes on Discovering Trading Strategies.

Fung, G. P., Yu, J. X., & Lu, H. (2005). The Predicting Power of Textual Information on Financial Markets. *IEEE Intelligent Informatics Bulletin 5.1*, 1-10.

Fung, G. P., Yu, J. X., & Lu, H. (2005). The Predicting Power of Textual Information on Financial Markets. *IEEE Intelligent Informatics Bulletin 5.1*, 1-10.

Gidoflavi, G., & Elkan, C. (2001). *Using news articles to predict stock price movements.* Department of Computer Science and Engineering, University of California, San Diego.

Golug, G., & Kahan, W. (1965). Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industria &l Applied Mathematics, Series B: Numerical Analysis, 2*(2), 205-224.

Guang, Q., Xiaofei, H., Feng, Z., Yuan, S., Jiajun, B., & Chun, C. (2010). DASA: dissatisfaction-oriented advertising based on sentiment analysis . *Expert Systems with Applications* , 6182-6191.

Hasson, M. R., & Baikunth, N. (2005). Stock market forecasting using Hidden Markov Model: A New Approach. *ISDA'05 Proceedings. 5th International Conference on. IEEE* (pp. 192-196). Intelligent Systems Design and Applications.

Kang, H., Yoo, S. J., & Han, D. (2012). Senti-lexicon and improved naive bayes algorithms for sentiment analysis of restaurant reviews. *Expert Systems with Applications* , 6000-6010.

Kaufmann, M. (2012). JMaxAlign: A Maximum Entropy Parallel Sentence Alignment Tool. *COLING (Demos)* , 277-288.

Kim, K.-j., & Han, I. (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with Applications, 19*(2), 125-132.

Kim, S., & Hovy, E. (2004). Determining the sentiment of opinions. *Proceedings of international conference on computtational linguistics* .

Lane, G., & Lane, C. (1998). *Getting Started with Stochastics.*

Lavrenko, V., Schmill, M., Lawrie, D., Ogilvie, P., Jensen, D., & Allan, J. (2000). Mining of concurrent text and time series. *KDD-2000 Workshop on Text Mining.*

Liu, B. (2012). Sentiment Analysis and Opinion Mining . *Synthesis Lectures on Human Language Technologies* , 1-167.

Martineau, J., & Finin, T. (2009). Delta TFIDF: An Improved Feature Space for Sentiment Analysis . *ICWSM.* .

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal, 5*(4), 1093-1113.

Minging, H., & Bing, L. (2004). Mining and summarizing customer reviews. *Proceedings of the ACM SIGKDD international conference on Knowledge Discovery and Data Mining*.

Mittermayer, M.-A., & Knolmayer, G. F. (2006). Newscats: A news categorization and trading system. *Data Mining, 2006. ICDM'06. Sixth international Conference on*, (pp. 1002-1007).

Moraes, R., Valiati, J. F., & Neto, W. P. (2013). Document-level sentiment classification: An empirical comparison between SVM and ANN. *Expert Systems with Applications 40.2*, 621-633.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification Using Machine Learning Techniques . *Proceedings of the ACL-02 conference on Empirical Methods in Natural Language Processing - Volume 10* (pp. 79-86). Associaion for Computational Linguistics.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. *Advances in large margin classifiers, 10*(3), 61-74.

Salton, G., Wong, A., & Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications fo the ACM, 18*(11), 613-620.

Schumaker, R. P., & Chen, H. (2005). A Quantitative Stock prediction System based on Financial news. *Information Processing & Management, 45*(5), 571-583.

Strzalkowski, T. (1994). Document representation in natural lnaguage text retrieval. *Proceedings of the workshop on Human Language Technology* (pp. 364-369). Association for Computational Linguistics.

The Stanford NLP(Natual Language Processing) Group. (2015, January 29). Retrieved April 2, 2015, from The Stanford NLP(Natual Language Processing) Group: nlp.standford.edu/softwaretagger.shtml

Thomas, J. D., & Sycara, K. (2000). Integrating genetic algoirthms and text learning for financial prediction. *Data Mining with Evolutionary Algorithms*, 72-75.

Turney, P. D. (2002). Thumbs up or thumbs down?: sematic orientation applied to unsupervised classification of reviews. *Proceedings of the 40th annual meeting on association for computational lingusitics* (pp. 417-424). Association for Computational Linguisitcs.

Turney, P. D., & Littman, M. L. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems(TOIS) 21.4* , 315-346.

Whitelaw, C., Garg, N., & Argamo, S. (2005). Using appraisal groups for sentiment analysis. *Proceedings of the 14th ACM international conference on Information and knowledge management* (pp. 625-631). ACM.

Wilson, T., Wiebe, J., & Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. *Proceedings of the conference on human language technology and empirical methods in natural language processing* (pp. 347-354). Association for Computational Linguistics .

Yong, Y., Xu, C., & Ren, G. (2011). Sentiment Analysis of Text Using SVM. *Electrical, Information Engineering and Mechatronics*, 1133-1139.

Zhan, J., Cohen, N., & Atreya, A. (n.d.). Sentiment Analysis of News Articles for Financial Signal Prediction .

Zhang, W., & Skiena, S. (2010). Trading strategies to exploit blog and news sentiment. *ICWSM*.

Zhang, Y. (2004). *Prediction of finacnial time series with Hidden markov Models.*